SourceGate® II

User's Guide

for the

HMI Background Mode Debugger (BMD)

Version 3.0.0
March 3, 1997

Huntsville Microsystems, Inc. © 1997

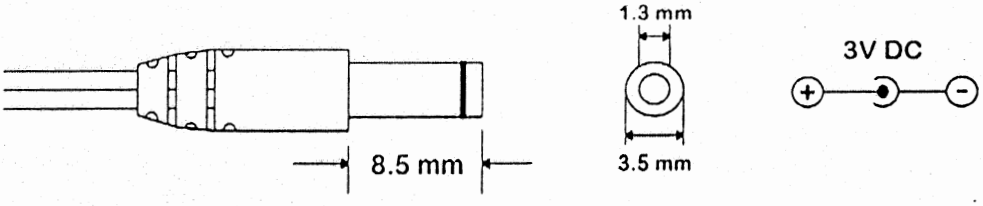
Supplying Power to the HMI BMD

New versions of the HMI BMD provide the option to obtain power from either an external power supply or from the target system via the BDM connector. The default configuration is to obtain power from an external supply, and HMI includes a 110Vac to 3Vdc wall transformer with the BMD. For users in areas where 110Vac is not available, a power supply which meets the following specifications may be used:

Voltage: 3V DC
Current: ~300mA
Connector: 3.5mm outside (negative) x 1.3mm female inside (positive) x 8.5mm length

To reconfigure the BMD to obtain power from the target system, open the BMD's box by inserting a small screwdriver in the slots on each side of the box and gently prying the top and bottom apart. Once open, locate the three-pin header labeled "P8" and move the shorting block so that it connects the center pin labeled "PWR to the pin labeled "TAR". To switch back to external power, simply move the shorting block so that it connects the center pin labeled "PWR" to the pin labeled "ACA".

Note that some target systems have a current-limiting resistor placed in the circuit between the board's power supply and the Vdd pin of the BDM connector. This can prevent the BMD from running properly. In this case the user must either short the resistor, or use an external power supply.



SourceGate® II

User's Guide

for the

HMI Background Mode Debugger (BMD)

Version 3.0.0
March 3, 1997

Huntsville Microsystems, Inc. © 1997

How To Use This Manual

The *HMI Development Tools* manual consists of the following sections:

Introduction
The Introduction section provides you with the latest Release Notes and other information to help you become familiar with your new system.

User's Guide
The SourceGate II User's Guide is included in this section. This User's Guide will provide detailed information on how to use SourceGate II with your system. Chapters 1, 2, 3, 5, 6, and the Appendix are common to all HMI systems. Chapter 4, along with the Table of Contents and Index sections, are unique to your HMI system. If you have trouble finding information in this User's Guide, consult the on-line help or contact HMI Technical Support.

Appendix
The Appendix section will contain your Software, License Agreement, Software Registration card, and any Application Notes for your system.

If you find any errors, or if you have a suggestion on how to improve this manual, please write us or contact HMI Technical Support at 205-881-6005 or via e-mail at support@hmi.com.

Thank you for purchasing HMI products.

Huntsville Microsystems, Inc. © 1997
P.O. Box 12415
Huntsville, AL 35815

256-881-6022 x400 Support
256-881-6005 Sales
256-882-6701 Fax

sales@hmi.com (Sales)
support@hmi.com (Technical Support)
http://www.hmi.com/ (WWW Site)
ftp://ftp.hmi.com/ (FTP Site)

Need Help?

Call Us!

If you experience difficulty with this product, please call our TECHNICAL SUPPORT LINE:

256-881-6022 x400 Support
256-881-6005 Sales
256-882-6701 Fax

You may also contact us via e-mail at:

support@hmi.com

or transfer files to our FTP server at:

ftp.hmi.com

We also maintain a Bulletin Board which may be used for file transfer purposes at:

(205) 881-7395

Keep in touch with the latest HMI product offerings via our World Wide Web site at:

www.hmi.com

HUNTSVILLE MICROSYSTEMS, INC.
P.O. Box 12415
Huntsville, AL 35815

SourceGate II Version 3.0.0
Release Notes

1.0 Purpose

This document outlines relevant information including any known problems in this release of SourceGate II.

If you have any questions or problems, please contact HMI Technical Support at (205) 881-6005 or via Internet E-mail at the address 'support@hmi.com'.

2.0 Using The HMICOMM.DRV Driver (Not Applicable For BMDs and Simulators)

During the installation process **(under Windows 3.1x only)**, the installer will ask if you want to install the HMI Communications Driver 'hmicomm.drv' into your 'system.ini' file. This driver replaces the standard Windows serial port driver, 'comm.drv'. If you elect not to install this driver, the installer will print a warning message that baud rates greater than 9600 can cause communications problems between SourceGate II and the HMI emulator. There are a couple of additional things to keep in mind:

1) If you do allow the installer to install the 'hmicomm.drv' driver, this driver replaces the existing driver in the "comm.drv=" statement in the [boot] section of your 'system.ini' file. You will have to restart Windows to load the driver for use. Additionally, if another application had previously modified the "comm.drv=" statement with a driver of its own, that application's driver will no be longer active. This may cause problems when running the other application. (Your original serial port driver is left unmodified, should you need to reinstall it.)

2) If you do not allow the installer to install the 'hmicomm.drv' driver, you will probably experience communications problems with the emulator when running at baud rates higher than 9600 (this is a general Windows problem, and whether you experience it or not is dependent on your computer's exact hardware and configuration). If you experience problems communicating at higher baud rates, look for the 'hmicomm.drv' driver in the "comm.drv=" statement in the [boot] section of your 'system.ini' file. If this driver is not loaded, you will need to modify this line by hand to load the driver. 'Hmicomm.drv' is placed by the installer in the SourceGate II installation directory for your convenience should you need to use the driver at a later date.

Note: This driver is not required and will not be installed when running SourceGate II under Windows 95, Windows NT, or Unix.

3.0 Communicating To Emulators Via Ethernet (Requires Emulator w/Ethernet Option)

The following sections discuss how to communicate via Ethernet to those HMI emulators that are equipped with the Ethernet option.

3.1 Windows 3.1x/95/NT

Using an Ethernet emulator requires your PC to have Ethernet hardware and a compatible 'winsock.dll' file. SourceGate II has been tested with the 'winsock.dll' files from Microsoft Windows 95, Novell LAN WorkPlace, and FTP Software. Any 'winsock.dll' compliant with Version 1.1 of the Windows Sockets standard should work, but there are non-standard Winsocks available from other vendors that may result in some Winsock/SourceGate II combinations being unreliable.

A Windows Sockets 1.1 compatible TCP/IP stack will typically have a 'hosts' file that may be configured with an emulator name that you select and the Ethernet IP address that you assigned the emulator. The entry in the 'hosts' file should look something like this:

```
192.9.200.201  HMI_Emulator
```

(Note that you will need to know the IP address of the emulator and substitute that address for the address "192.9.200.201" given in the above example.)

You can now invoke SourceGate II and communicate with your Ethernet emulator by setting the Windows program item properties command line option to the following:

```
SG2 -iHMI_Emulator
```

You can also just enter the IP address on the command line as in the following example:

```
SG2 -i192.9.200.201
```

If your Windows TCP/IP software includes 'ping' and 'finger' utilities, you can 'ping' the emulator to see if it is alive and you can 'finger' the emulator to see who is using it:

```
ping HMI_Emulator
finger @HMI_Emulator
```

3.2 Unix Systems

To set your system up for use with an Ethernet emulator (requires an emulator with the Ethernet option), please perform the following steps:

1) Become Superuser (Note that 'su' will require the Superuser password for access):

```
su
```

2) Edit the file '/etc/hosts' to add the IP address and host name for your Ethernet emulator:

```
vi /etc/hosts
```

Use the 'G' command from within 'vi' to go to the last line of the file. Use the 'o' command to start a new line. Type the new line, which will look something like this:

```
192.9.200.201 HMI_Emulator
```

Note that you will need to know the IP address of the emulator and substitute that address for the address "192.9.200.201" given in the above example.

Use the 'ESC' key to get 'vi' out of insert mode. Save your edits and get out of 'vi' with the ':x' command.

3) Stop being Superuser:

```
exit
```

You can now invoke SourceGate II and communicate with your Ethernet emulator by entering the following command:

```
sg2 -iHMI_Emulator
```

Note that you can also 'ping' the emulator to see if it is alive, and you can 'finger' the emulator to see who is using it:

```
ping HMI_Emulator
finger @HMI_Emulator
```

4.0 New Features

The following new features have been added in v3.0.0 of SourceGate II. Reference the *SourceGate II User's Guide* or see the on-line help for additional information on these new features.

SourceGate II Menu Enhancements

SourceGate II menus have been restructured to make the menu layout more logical and intuitive. The Source menu item was renamed to CodeView and a Run menu item was added. A few menu items were moved to place them under a more appropriate menu heading.

Accelerator Key Changes

A few SourceGate II accelerator keys have been changed to accommodate the new CodeView windows and new ones have been added to allow keyboard access to basic emulator execution functions (Run/Stop/Reset).

Enhanced CodeView Window

The CodeView window has been completely redesigned to implement requested features and improve performance. Vertical scrolling speed has been greatly enhanced. New CodeView preferences allow the user to specify colors for the code text, program counter, breakpoints set, and background. The new CodeView module windows can show source code only, assembly only, or source and assembly code interleaved. A newly designed VASM window now has the capability of decreasing the windows starting address without having to open up a new window.

pSOS Support

The display of pSOS data is now supported.

Multiple Trace Windows

In those units that support binary (fast) trace, multiple trace windows can be shown at any time.

68060 Auto Opcode Tagging

Automatic opcode trace tagging in those 68060 emulators that support this feature is now available. This feature allows SourceGate II to automatically opcode tag the trace buffer without the user having to perform the "OP" command from the Command window.

5.0 Known Problems In This Release

The following problems are known to exist in this version of SourceGate II.

5.1 Known Problems Affecting All Platforms

The following are known problems in this version of SourceGate II that are independent of the host platform:

Compiler Interface Options

Changing options (byte order, etc.) in the Compiler Interface ("Config/Compiler Interface..." menu item) dialog will not immediately update any open windows which are affected by the change. The change for a window will take effect the next time the window is updated with new information.

For example, if the byte order is changed and the Watch window is on-screen, the new byte order will not effect the variables displayed in the Watch window until a Watch window update occurs.

Double And Long Double Values In The Watch Window

If the value of double or long double variables in the Watch window are not correct, the most probable cause is a reversed byte order in the Compiler Interface dialog. Go to the Compiler Interface dialog (choose 'Config/Compiler Interface...' from the main SourceGate II menu), and reverse the byte order strings for the Double and Long Double entries. Dismiss the dialog and on the next Watch window update the values for the double and long double variables should be correct.

Resetting Ethernet Emulators

Selecting 'Config/Reset Serial Port' when using an Ethernet emulator causes the Ethernet emulator to drop the connection.

5.2 Known Problems Affecting The Windows Platform

The following are known problems in this version of SourceGate II that only affect the Microsoft Windows platform:

INI File

Whenever you exit from SourceGate II, the windows that are open, along with their sizes, positions on screen, and other relevant information, are saved in a file named '<EmulatorName>.ini', where <EmulatorName> is the name of the emulator that you are using (e.g. '68000.ini', '8051.ini', etc.). When SourceGate II is restarted later, it will reference this file and reopen all of the windows that were active when you last exited, placing them at their previous locations on screen. Under normal circumstances this works well without any problems. However, should you change your Windows screen resolution, your SourceGate II windows may be reopened at odd locations and/or with unusable sizes. As a precaution, whenever you change your Windows screen resolution you should delete the SourceGate II '<EmulatorName>.ini' file.

Note, however, that deleting the SourceGate II '<EmulatorName>.ini' file will also destroy any saved preferences settings.

5.3 Known Problems Affecting The SUN/SPARC Platform

The following are problems that have been observed on the SUN/SPARC platform:

Printing

In order to print, you must first print to a file. You may then print the file to your printer.

Iconizing A Window

If a window is iconized, the only way to reopen the iconized window is by clicking on the icon. The window cannot be reopened by selecting it from the open windows list.

Monochrome Monitors

On systems using monochrome monitors, some windows will display a black cursor on a black background, rendering the cursor difficult to see. This occurs most notably in the "Config/Emulator Memory Map..." window.

Help and Modal Dialogs

The menu, buttons, and cross-references in the Help window do not work if Help is brought up over a Modal dialog (such as the Event Address Binary or Watch Entry dialogs). The Modal dialog behind the Help window must be dismissed before the Help window controls will work.

"Config" Can Take Over Two Minutes On Ethernet Emulators

The "config" command that restores the emulator memory map, sequences, and other battery backed-up configuration values to the factory configuration can take over two minutes on some emulator models when these units are connected via Ethernet to the host workstation. After typing "config" in the Command window, SourceGate II will appear to be locked up until the "config" command is completed.

Num Lock

On some systems, when "Num Lock" is lit (selected), the space bar no longer works in any text edit box. This problem has been noted on a SPARCStation IPX with a Type 5 keyboard and on a SPARCStation IPC with a Type 4 keyboard, both running SunOS 4.1.3. It has not been a problem on a SPARCStation LX running SunOS 5.3 with a Type 5 keyboard.

Warnings out of 'ld.so'

SourceGate II for SUN was compiled on a Sun SPARCStation running SunOS 4.1.3_U1. If your version of SunOS does not match 4.1.3_U1, then you may receive the following warning message each time SourceGate II is started:

ld.so: warning:/usr/lib/libc.so.1.x has older revision than expected x

This warning message is generated because your version of the shared library 'libc' does not match the version expected by SourceGate II. It has been our experience that this message is annoying but harmless. The functions used by SourceGate II from this shared library are unchanged from previous versions of libc and, therefore, older versions of libc may be safely used by SourceGate II.

5.4 Known Problems Affecting The HP Platform

The following are problems that have been observed on the HP platform:

Printing

In order to print, you must first print to a file. You may then print the file to your printer.

Iconizing A Window

If a window is iconized, the only way to reopen the iconized window is by clicking on the icon. The window cannot be reopened by selecting it from the open windows list.

Monochrome Monitors

On systems using monochrome monitors, some windows will display a black cursor on a black background, rendering the cursor difficult to see. This occurs most notably in the "Config/Emulator Memory Map..." window.

Help and Modal Dialogs

The menu, buttons, and cross-references in the Help window do not work if Help is brought up over a Modal dialog (such as the Event Address Binary or Watch Entry dialogs). The Modal dialog behind the Help window must be dismissed before the Help window controls will work.

"Config" Can Take Over Two Minutes On Ethernet Emulators

The "config" command that restores the emulator memory map, sequences, and other battery backed-up configuration values to the factory configuration can take over two minutes on some emulator modules when these units are connected via Ethernet to the host workstation. After typing "config" in the Command window, SourceGate II will appear to be locked up until the "config" command is completed.

Menu Bar Resizing

When SourceGate II is started, the main menu bar will shrink and resize itself several times. This behavior is part of the normal startup sequence and is not a bug.

6.0 Command (.CMD) Files

Included on the installation disk is a file called 'demo.cmd'. This file will be copied by the installer to your installation directory. This command file will perform certain tasks, such as defining reset and stack pointer vectors, reading the SourceGate II demo file, and resetting the system. If you allow the installer to start SourceGate II during the install process, this command file will be run automatically for you. From the SourceGate II menu bar, you can use 'File/Record...' to record a command file and 'File/Execute...' to execute a previously recorded command file. **Note: Command files only work in the Command window. They cannot be used to record operations in other windows.**

7.0 The Performance Analysis Card (PAC) (N/A For BMDs)

If your system has the PAC option installed (or if you are using a Simulator), there is a file included with the SourceGate II installation named 'pac.set' which will allow you to experiment with the PAC system while running the supplied SourceGate II demo files. To load this 'pac.set' file, choose 'Display/Performance Analysis' from the SourceGate II menu bar (to enable PAC), then choose 'PAC/Restore PAC Setup...', and enter the path of the supplied 'pac.set' file. Once loaded, this file will give histograms and data that is valid for the demo program. You can create your own 'pac.set' files with the 'PAC/Save PAC Setup...' menu option and use the 'PAC/Restore PAC Setup...' menu option to reload them later. **Note: You must have the PAC option installed on your system for this to work.**

8.0 Problem Fixes

For a list of problems that were present in version 2.0.1 of SourceGate II which have been fixed in version 3.0.0, please see the SourceGate II help file under the section "Release Notes/Problem Fixes".

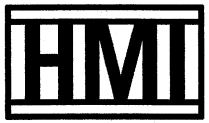
SourceGate® II

User's Guide

for the

HMI Background Mode Debugger (BMD)

Version 3.0.0
March 3, 1997



Huntsville Microsystems, Inc. © 1997

The contents of this document are believed to be accurate. However, Huntsville Microsystems, Inc. assumes no responsibility for the use of this document and issues no warranties, implied or otherwise, on the contents herein. Furthermore, Huntsville Microsystems, Inc. reserves the right to revise this publication and to make changes to the contents, without obligation to notify any persons of such revisions or changes.

Copyright © 1997 by Huntsville Microsystems, Inc.

SourceGate is a registered trademark of Huntsville Microsystems, Inc.
Sun and Sun logo are trademarks of Sun Microsystems, Inc.
HP and Hewlett-Packard logo are trademarks of Hewlett-Packard, Inc.
Pentium is a registered trademark of Intel Corporation.
Microsoft is a registered trademark and Windows is a trademark of Microsoft Corporation.

This document was created entirely with AmiPro, which is a trademark of Lotus Development Corporation.

Table Of Contents

Chapter 1 - Getting Started 1-1

Checking Your Package 1-1

Disks 1-1

Documentation 1-1

Hardware 1-1

SPS-2000 Series Emulators 1-1

HMI-200 Series Emulators 1-1

Background Mode Debuggers (BMDs) 1-1

Simulators 1-1

System Requirements 1-2

Hardware 1-2

Software 1-2

Memory 1-2

Disk Space 1-2

Installing SourceGate II 1-3

Installing The Windows Version Of SourceGate II 1-3

Installing The Sun Version Of SourceGate II 1-3

Installing The HP Motif Version Of SourceGate II 1-4

Hardware Connections 1-5

HMI Hardware To Computer Connection 1-5

SPS-2000 Series Emulators 1-5

Serial Connection 1-5

Ethernet Connection 1-5

Parallel Connection 1-6

HMI-200 Series Emulators 1-6

Serial Connection 1-6

Ethernet Connection 1-6

Table Of Contents (Cont.)

Parallel Connection 1-6
Background Mode Debuggers (BMDs) 1-6
Serial Connection 1-6
Simulators 1-6
HMI Hardware To Target System Connection 1-7
Starting SourceGate II 1-7
Starting The Windows Versions Of SourceGate II 1-7
Windows 3.1x 1-7
Windows 95 and Windows NT 1-8
Starting The Unix Versions Of SourceGate II 1-8
Command Line Options 1-9
Environment Variables 1-13

Chapter 2 - General Overview 2-1

Becoming Familiar With The SourceGate II Window 2-1
Control Boxes 2-1
Menu Bar 2-1
Title Bar 2-2
Restore Button 2-2
Maximize Button 2-2
Minimize Button 2-2
Scroll Bars (Horizontal and Vertical) 2-2
Scroll Boxes (Horizontal and Vertical) 2-2
SourceGate II Menu Bar 2-2
The 'File' Menu 2-2
Download Object File... 2-2
Upload Object File... 2-2
Execute... 2-2

Table Of Contents (Cont.)

Record... 2-3
Log... 2-3
Print 2-3
Printer Setup... 2-3
Exit SourceGate... 2-3
The 'Edit' Menu 2-3
The 'Setup' Menu 2-3
The 'Display' Menu 2-3
Command Window 2-3
Performance Analysis 2-3
Trace 2-4
Watch 2-4
Watch In Real Time 2-4
Registers 2-4
Memory 2-4
Stack Trace 2-4
Message Window 2-4
C++ Class Browser 2-4
pSOS Data 2-4
The 'CodeView' Menu 2-4
Modules... 2-4
Symbols... 2-5
List BPs 2-5
View PC 2-5
VASM 2-5
PC Tracking Module 2-5
Cursor Tracking Assembly 2-5

Table Of Contents (Cont.)

Start 2-5
Stop 2-5
Reset 2-5
Step In 2-6
Step Over 2-6
The 'Run' Menu 2-6
Emulator Status 2-6
Go From/To... 2-6
Go 2-6
Stop 2-6
Reset 2-6
The 'Config' Menu 2-7
Preferences 2-7
Register Map 2-7
Compiler Interface... 2-7
Emulator Memory Map... 2-7
Simulator 2-7
Serial Interface... 2-7
Restore Memory Map... 2-7
Save Memory Map As... 2-7
Factory Configuration... 2-7
Reset Serial Port 2-7
The 'Trace' Menu 2-8
Freeze 2-8
Resume 2-8
Break Buffer 2-8
Trigger Buffer 2-8

Table Of Contents (Cont.)

View 2-8
Timing 2-8
Goto Top 2-8
Goto Bottom 2-8
Find Trigger 2-8
Ports 2-8
Filter 2-8
The 'PAC' Menu 2-9
Profile Setup... 2-9
Profile Data... 2-9
Profile Graph... 2-9
Coverage 2-9
Extended Coverage 2-9
Extended Source 2-9
Restore PAC Setup... 2-9
Save PAC Setup... 2-9
Close PAC 2-9
The 'Window' Menu 2-9
Next 2-9
Previous 2-10
Cascade 2-10
Tile Horizontal 2-10
Tile Vertical 2-10
Close 2-10
Close All 2-10
Open Window List 2-10
The 'Help' Menu 2-10

Table Of Contents (Cont.)

On Context 2-10
On Window 2-10
Keyboard 2-10
Index 2-11
How Do I... 2-11
Help On Help 2-11
Search 2-11
About 2-11
Window Manipulation 2-11
Changing To The Next Window 2-11
Changing To The Previous Window 2-11
Choosing From The Open Window List 2-11
Cascading All Open Windows 2-12
Closing The Current Active Window 2-12
Closing All Open Windows 2-12
Defining Your Printer 2-12
Working With Symbols 2-12
Symbol Selection Window 'Modules' Field 2-13
Symbol Selection Window 'Symbols' Field 2-13
Symbol Selection Window Buttons 2-13
Using Help 2-15
Context Sensitive Help 2-15
Active Window Help 2-15
Help Menu 2-15
On Context 2-15
On Window 2-15
Keyboard 2-15

Table Of Contents (Cont.)

Index 2-16
How Do I... 2-16
Help On Help 2-16
Search 2-16
About 2-16
Exiting SourceGate II 2-17
Accelerator Keys 2-17
Setting Compiler Interface Options 2-19
Line Symbol Format Field 2-19
Default Source Ext Field 2-20
Program Counter Field 2-20
Stack Pointer Field 2-20
Stack Frame Pointer Field 2-20
Floating Point Format Field 2-20
Type/Byte Order Fields 2-20
Pointers/Near/Far Fields 2-20
Floating Point Format Fields 2-20
Compiler Interface Options Window Buttons 2-20
Viewing Serial Interface Parameters 2-21
Interface Window Buttons 2-22
Configuring User-Defined Preferences 2-22
File Loading Preferences 2-22
File Load Preferences Window Buttons 2-23
CodeView Window Preferences 2-23
INI File Settings 2-23
Working With Object Files 2-24
Downloading An Object File 2-25

Table Of Contents (Cont.)

File Down Load Options Window Fields 2-26
File Download Options Window Buttons 2-27
Uploading An Object File 2-28
File Upload Options Window Fields 2-28
File Upload Options Window Buttons 2-29
Chapter 3 - Using SourceGate Windows 3-1
Using The Command Window 3-1
Opening The Command Window 3-2
Entering Command Window Commands 3-2
Available Command Window Commands 3-2
Command Window Buttons 3-2
Command Line Overrides 3-3
Using Symbols In The Command Window 3-3
Using Module Commands In The Command Window 3-5
Setting Up Module Search Paths and Source File Names 3-6
Creating Automated Command Files From The Command Window 3-8
Using The Emulator Status Window 3-9
Opening The Emulator Status Window 3-9
Emulator Status Window Border 3-9
Emulator Status Window 'Status' Field 3-9
Emulator Status Window 'Trace' Field 3-10
Emulator Status Window 'Pass Count' Fields 3-10
Emulator Status Window 'Time/Count' Fields 3-10
Emulator Status Window 'Timer' Field 3-10
Emulator Status Window Buttons 3-10
Using The Memory Window 3-12
Opening The Memory Window 3-12

Table Of Contents (Cont.)

Memory Window Buttons 3-13
Viewing Specific Memory Locations 3-15
Memory Goto Window 3-15
Memory Goto Window Buttons 3-15
Modifying Specific Memory Locations 3-16
Changing The Memory Window Display Mode 3-17
Memory Display Format Window 3-17
Memory Display Format Window Buttons 3-18
Searching Memory For A Specific Value 3-19
Memory Search Window 3-19
Memory Search Window Buttons 3-20
Comparing Memory Blocks 3-21
Memory Compare Window 3-21
Memory Compare Window Buttons 3-22
Filling A Memory Block With A Constant Value 3-23
Memory Fill Window 3-23
Memory Fill Window Buttons 3-24
Moving A Memory Block 3-25
Memory Move Window 3-25
Memory Move Window Buttons 3-26
Testing A Memory Block 3-27
Memory Test Window 3-27
Memory Test Window Buttons 3-28
Accessing Memory Mapped I/O Locations 3-29
Memory I/O Window 3-29
Memory I/O Window Buttons 3-30
Memory Output Window 3-31

Table Of Contents (Cont.)

Memory Output Window Buttons 3-32
Using The Watch/Watch In Real Time Windows 3-33
Opening The Watch/Watch In Real Time Windows 3-34
Watch/Watch In Real Time Window Buttons 3-34
Adding a Watch Variable 3-35
Entering Watch Variables Manually In The Watch Window 3-35
Watching a Symbol 3-35
Watch Variable Types 3-36
Watch Variable Overrides 3-37
Watch Variable Addresses 3-37
Watch Variable Display Types 3-37
Using The Watch Entry Window To Enter Watch Variables 3-38
Watch Entry Window Buttons 3-38
Deleting a Watch Variable 3-43
Modifying Watch Window Variable Values 3-43
Saving a Watch Window 3-43
Restoring a Saved Watch Window 3-43
Watch Window Examples 3-43
Watching A Structure 3-43
Watching An Element Of A Structure 3-44
Watching A One-Dimensional Array 3-44
Watching A Multi-Dimensional Array 3-44
Watching Pointers 3-45
Using Overrides In The Watch Window 3-45
Using The Trace Window 3-46
Opening The Trace Window 3-46
Capturing Trace Data 3-47

Table Of Contents (Cont.)

Choosing Between Break and Trigger Buffers 3-47
Description of Trace Display Fields 3-48
Trace Cycle Field 3-48
Trace Bank Field 3-49
Trace Address Field 3-49
Trace Data Field 3-49
Trace External Field 3-49
Trace Status Bit Field 3-49
Trace Bus Activity Field 3-50
Trace Time Stamp Field 3-50
Trace 'Ports' Switch 3-50
Trace 'Filtered' Switch 3-50
Formatting the Trace Display 3-51
Trace Disassembly Format 3-51
Trace State Format 3-51
Trace Source Format 3-51
Trace Disassembly and State Format 3-51
Trace Disassembly and Source Format 3-52
Trace Disassembly, State, and Source Format 3-52
Trace Timing Off Format 3-52
Trace Timing Relative Format 3-52
Trace Timing Delta Format 3-52
Using the Freeze Trace Feature 3-53
Trace 'Goto' Commands 3-53
Trace Search Window 3-54
Trace Search Window Buttons 3-54
Saving Trace Information to a File 3-55

Table Of Contents (Cont.)

Print Trace Buffer Window 3-55
Print Trace Buffer Window Buttons 3-55
Trace Window Buttons 3-56
Using The Register Window 3-57
Opening The Register Window 3-58
Modifying Register Values 3-58
Available Registers 3-58
Using The 'Go From/To...' Window 3-59
'Go From/To...' Window 'Go From Address' Field 3-59
'Go From/To...' Window 'Go To Breakpoints' Field 3-59
'Go From/To...' Window Buttons 3-60
Using The Stack Trace Window 3-61
Opening The Stack Trace Window 3-61
Stack Trace Window Contents 3-61
Using The Register Map Window 3-63
Opening The Register Map Window 3-63
Register Map Window Buttons 3-63
C++ Class Browser/Sub Browser Windows 3-64
Opening The C++ Class Browser/Sub Browser Windows 3-64
C++ Class Browser Window 3-64
C++ Class Sub Browser Window 3-65
Using The pSOS Window 3-67

Chapter 4 - The BMD Interface 4-1

BMD Command Window 4-1
Command Line Overrides 4-2
S (Supervisor Mode 4-2
U (User Mode) 4-2

Table Of Contents (Cont.)

P (Program Mode) 4-2
D (Data Mode) 4-2
L (Long Word Mode) 4-2
W (Word Mode) 4-3
B (Byte Mode) 4-3
Available Command Window Commands 4-4
A (In-Line Assembly) Command 4-5
BM (Byte Mode) Command 4-5
C (Compare Memory Block) Command 4-6
CM (Show Status Of Control Board Memory) Command 4-7
D (Dump Memory Block) Command 4-7
DM (Data Memory Mode) Command 4-7
E/EN (Enter/Change Memory) Command 4-8
EXP (Examine/Change Peripheral) Command 4-9
EXR (Examine/Change Peripheral Register) Command 4-9
F (Fill Memory Block) Command 4-10
G (Begin Real-Time Emulation) Command 4-11
HLD (Hex Listing Disabled) Command 4-12
HLE (Hex Listing Enabled) Command 4-12
I/IN (Input) Command 4-12
L (List Disassembled Code) Command 4-13
LM (Long Word Display Mode) Command 4-13
M (Move Memory Block) Command 4-14
MT/MTL (Test Block Of Memory) Command 4-15
O/ON (Output) Command 4-16
PM (Program Memory Mode) Command 4-16
RD (Read Object File Into Memory) Command 4-17

Table Of Contents (Cont.)

RP (Reset Processor, Continue Emulation) Command 4-18
RS (Reset Processor, Stop Emulation) Command 4-18
S_PMBA (Set Prog. Module Base Address) Command 4-19
S_ROMWIDTH (Set ROM Width) Command 4-20
SM (Supervisor Memory Mode) Command 4-24
SP (Stop Emulation) Command 4-24
SR/SRN (Search Block Of Memory) Command 4-25
SS (Single Step) Command 4-26
ST (Show Emulator Status) Command 4-28
UM (User Memory Mode) Command 4-28
VASM (View Assembly) Command 4-28
VER (Display Firmware Version) Command 4-28
W (Write Memory Contents To File) Command 4-29
WM (Word Display Mode) Command 4-30
X (Examine/Change Registers) Command 4-30
The BMD 'Setup' Menu 4-31
Using The Register Window 4-31
Opening The Register Window 4-32
Modifying Register Values 4-32
Available Registers 4-32
Chapter 5 - The CodeView Interface 5-1
What Is The CodeView Interface? 5-1
CodeView Window Fields 5-2
CodeView Window Buttons 5-5
Setting CodeView Window Preferences 5-7
CodeView Setup Window 5-7

Table Of Contents (Cont.)

CodeView Setup Window Buttons 5-11
Codeview Global View Parameters Window 5-12
CodeView Global View Parameters Window Buttons 5-15
CodeView Color Selection Window 5-16
CodeView Color Selection Window Buttons 5-17
CodeView Functional Options Window 5-18
CodeView Functional Options Window Buttons 5-20
Loading Source Code Into CodeView Windows 5-21
Module Selection Window 5-21
Viewing Multiple CodeView Windows 5-22
Viewing Assembly Code In CodeView Windows 5-23
Changing The Display Mode In CodeView Windows 5-25
Starting Emulation From CodeView Windows 5-28
Setting Breakpoints In CodeView Windows 5-28
CodeView BPs Window 5-29
CodeView BPs Window Buttons 5-30
Single-Stepping In CodeView Windows 5-31
Using The PC Tracking Window 5-32
Using The Cursor Tracking Assembly Window 5-33
Searching The CodeView Window 5-34
CodeView Find Window Buttons 5-36

Chapter 6 - Performance Analysis (PAC) 6-1

What Is Performance Analysis? 6-1
Opening PAC Windows 6-2
PAC Setup 6-3
PAC Profile Setup Window 6-3
PAC Profile Setup Window Fields 6-4

Table Of Contents (Cont.)

SETUP Radio Button 6-4
MODE Radio Button 6-4
QUAL Radio Button 6-4
Num Field 6-4
Profile Module 6-4
Address Field 6-4
Range Field 6-5
Entry Field 6-5
Exit Field 6-6
Active Field 6-7
Symbol Lookup Field 6-7
PAC Profile Setup Window Buttons 6-7
PAC Profile Setup Window Example 6-8
PAC Profile Mode Setup Window 6-12
PAC Profile Mode Setup Window Fields 6-12
SETUP Radio Button 6-12
MODE Radio Button 6-12
QUAL Radio Button 6-12
Record Trigger Mode Field 6-13
Start Record Module Field 6-13
Stop Record Module Field 6-13
Record Time Field 6-13
Select Modules For New Coverage Field 6-13
PAC Profile Mode Setup Window Buttons 6-14
PAC Setup Qualify Window 6-15
PAC Setup Qualify Window Fields 6-15
SETUP Radio Button 6-15

Table Of Contents (Cont.)

MODE Radio Button 6-15
QUAL Radio Button 6-16
'Qualify' Check Box 6-16
'Read/Write' Field 6-16
'Program/Data' Field 6-16
PAC Setup Qualify Window Buttons 6-17
Saving and Restoring PAC Setups 6-18
Saving A PAC Setup 6-18
Restoring A Previously Saved PAC Setup 6-18
Recording PAC Profile Data 6-19
PAC Execution Profile Window 6-19
PAC Execution Profile Window Fields 6-19
PAC Execution Profile 'Cumulative' Field 6-19
PAC Execution Profile 'Runs Requested' Field 6-20
PAC Execution Profile 'Runs Executed' Field 6-20
PAC Execution Profile 'Recording Time' Field 6-20
PAC Execution Profile 'Total Recording Time' Field 6-20
PAC Execution Profile 'Timer Resolution' Field 6-20
PAC Execution Profile 'Profile Module' Field 6-20
PAC Execution Profile 'Executions' Field 6-20
PAC Execution Profile 'Min. Time' Field 6-21
PAC Execution Profile 'Max. Time' Field 6-21
PAC Execution Profile 'Avg. Time' Field 6-21
PAC Execution Profile '% Time' Field 6-21
PAC Execution Profile Window Buttons 6-22
PAC Execution Profile Graph Window 6-23
PAC Execution Profile Graph Window Buttons 6-23

Table Of Contents (Cont.)

Starting A PAC Profile Session 6-24
Stopping A PAC Profile Session 6-24
Viewing PAC Coverage Data 6-25
PAC Coverage Profile Window 6-25
PAC Coverage Profile Window Buttons 6-26
Starting A PAC Coverage Profile Session 6-26
Stopping A PAC Coverage Profile Session 6-26
PAC Extended Coverage Window 6-27
PAC Extended Coverage Window Fields 6-28
PAC Extended Coverage Window Buttons 6-30
PAC Extended Source Coverage Window 6-31
PAC Extended Source Coverage Window Fields 6-33
PAC Extended Source Window Buttons 6-34
Printing PAC Information 6-35
Printing Files From The Main PAC Window 6-35
Printing The PAC Setup 6-35
Printing PAC Profile Data 6-36
Printing PAC Extended Coverage Data 6-37
Printing PAC Extended Source Data 6-38

Appendix A-1

Error Messages A-1
All Systems A-1
SPS-2000 Series/HMI-200 Series Emulators A-4
Background Mode Debuggers (BMDs) A-5
Using Compilers and Assemblers A-7
Available File Converters A-8

1 Getting Started

Checking Your Package

This section lists the disks, documentation, and hardware included in your SourceGate II package. Take time now to check the contents of your package. If your package is not complete, contact HMI.

Disks

Your SourceGate II package should contain two 3.5" disks labeled *SourceGate II*. These disks should be located in the back of your *HMI Development Tools* manual.

Documentation

Your package should contain the *HMI Development Tools* manual which provides a complete reference for your HMI system and the SourceGate II software that drives it. On-line help is also available for all systems via the SourceGate II Help menu.

Hardware

Your package should contain the following hardware:

SPS-2000 Series Emulators

SPS-2000 main unit, SPS pod, target interface pod, power cord, 25-pin RS-232 cable, 25-to-9 pin adapter, 25-to-phone jack (RS-232) adapter, phone jack (RS-232) cable, external trace bit leads, and Ethernet "flip cable".

HMI-200 Series Emulators

HMI-200 main unit, target interface pod, power cord, 25-pin RS-232 cable, 25-to-9 pin adapter, external trace bit leads, and Ethernet "flip cable" if the Ethernet option was ordered.

Background Mode Debuggers (BMDs)

BMD unit with debug port cable. You must supply a straight-thru 25-pin RS-232 cable.

Simulators

There is no hardware shipped with the HMI Simulator product.

System Requirements

This section describes the hardware, software, memory, and disk space requirements for using SourceGate II.

Hardware

SourceGate II requires the following hardware as a minimum:

- *A 386, 486, or Pentium™ based computer certified for use with Windows 3.1/95/NT, a SUN SPARC computer, or a HP 9000 Series 700 computer*
- *A two or three button mouse or trackball is required*
- *One 1.44MB 3.5" disk drive*
- *One RS232 serial port or Ethernet connection*
- *Hard disk drive with at least 4MB of free disk space*

Software

If running SourceGate II on a PC, you must have Microsoft Windows 3.1 with MS-DOS version 3.3 or higher, Windows 95, or Windows NT.

The Sun workstation version of SourceGate II must have Sun OS version 4.1.x (or higher) or Solaris version 2.x (or higher) with OpenWindows version 3.0 or higher.

The HP workstation version of SourceGate II must have HP-UX version 8.x (or higher) running the OSF/Motif GUI.

Memory

The Windows version of SourceGate II requires a minimum of 4MB of RAM with 8MB recommended. Memory should not be an issue on UNIX systems since these systems support virtual memory.

Disk Space

SourceGate II requires a minimum of 4MB of hard disk space on both Windows and UNIX systems.

Installing SourceGate II

This section will describe how to install SourceGate II on your PC or UNIX system.

Installing The Windows Version Of SourceGate II

To install the Windows version of SourceGate II, place the disk labeled "SourceGate II - Disk 1" in your 1.44MB 3.5" disk drive, open the Windows File Manager (Explorer for Windows 95 and Windows NT), make the drive where the floppy is located to be the active drive, and run the "SETUP.EXE" program. You should then follow the instructions in the setup program in order to install the SourceGate II files correctly.

A Program Group will be created with the SourceGate II icon shown in this group under Windows 3.1. Windows 95/NT will add a HMI program group under the Start menu. This HMI program group will contain the shortcut to the actual SourceGate II program.

Installing The Sun Version Of SourceGate II

To install the Sun version of SourceGate II, perform the following steps:

cd <to the directory where you want the "hmi/sg2/xxxxx" directory to reside, where xxxxx is the processor being installed (68332, 68340, etc.)>
<Insert Disk 1>
tar xvf /dev/fd0 (or your floppy drive name if different)
<Eject Disk 1 and Insert Disk 2>
tar xvf /dev/fd0
<Eject Disk 2>
cd hmi/sg2/xxxxx (where xxxxx is the processor that was installed)
./sg2install

After the above steps have been completed, copy the sg2APP.color (for color systems) or sg2APP.mono (for monochrome systems) file to the /usr/openwin/lib/app-defaults directory into a file called 'sg2'. Some examples follow:

For color systems, type:

cp sg2APP.color /usr/openwin/lib/app-defaults/sg2

For monochrome systems, type:

cp sg2APP.mono /usr/openwin/lib/app-defaults/sg2

If you are not using Sun OpenWindows, you may need to copy the file to the X-Windows app-defaults directory. For example, if this directory is named /usr/X11/lib/app-defaults/sg2, then the following command needs to be performed :

cp sg2APP.color(or mono) /usr/X11/lib/app-defaults/sg2

Be certain that the LD_LIBRARY_PATH environment variable is correct. You can check it with the command 'printenv'. If it is wrong, then you will need to modify it as follows:

setenv LD_LIBRARY_PATH /usr/openwin/lib (or the correct path is)

The file sg2.frl is a resource file. It must be in the current directory or in a directory in the PATH environment variable. The files <processor>.sg and <processor>.sg2 are system specific configuration files and <processor>.csc is a help file, where <processor> is the actual name of the system being installed (e.g. 68332.sg, 68040.sg2, etc.). All of these files must reside in the current directory or in a directory in your PATH environment variable.

Printing support is only available if you set the environment variable XVTPATH to the directory 'print' that is supplied on the SG2 disks. If you don't set this environment variable to this directory, any print operation will respond with an error message and exit. **Note that this version of SourceGate II only allows printing to a file.**

Installing The HP Motif Version Of SourceGate II

To install the HP Motif version of SourceGate II, perform the following steps:

cd <to the directory where you want the "hmi/sg2/xxxxx" directory to reside,
where xxxxx is the processor being installed (68332, 68340, etc.)>
<Insert Disk 1>
tar xvf /dev/floppy (or your floppy drive name if different)
<Eject Disk 1 and insert Disk 2>
tar xvf /dev/floppy
<Eject Disk 2>
cd hmi/sg2/xxxxx (where xxxxx is the processor that was installed)
./sg2install

The file sg2.uid is a resource file. It must be in the current directory or in a directory in the UIDPATH. For example, if you wish to use the directory /usr/local/lib/sg2 to hold the UID file for SourceGate II (and for any other applications), you might include this line in your .cshrc file:

```
setenv UIDPATH /usr/local/lib/sg2/%U
```

or these lines in your .profile:

```
UIDPATH=/usr/local/lib/sg2/%U
export UIDPATH
```

The files <processor>.sg and <processor>.sg2 are system specific configuration files and <processor>.csc is a help file, where <processor> is the actual name of the system being installed (e.g. 68332.sg, 68040.sg2, etc.). All of these files must reside in the current directory or in a directory in your PATH environment variable.

Hardware Connections

This section will describe how to attach HMI hardware to your computer and target system.

HMI Hardware To Computer Connection

Before starting SourceGate II, you must connect your computer system to the HMI hardware. This section will explain how to do this.

SPS-2000 Series Emulators

The SPS-2000 Series emulators can be connected via one of two methods:

Serial Connection

To use a RS-232 serial communications link, you must connect a serial cable from a serial port of the computer to the RS-232 serial port of the emulator. This should be a "straight-thru" or "modem" type cable. Speeds of up to 57.6K baud are supported on most Pentium computers with 16550 UARTs running Windows 95 or Windows NT.

Ethernet Connection

If you are using the Ethernet connection, make sure that your network cable is properly fastened to either the BNC or phone jack connections on the back of the emulator. Also, before the emulator can be accessed via Ethernet, it will need to be configured for Ethernet operation. For additional information on how to set up an emulator for Ethernet operation, see the HMI Application Note *"ACEE-A-100: Configuring An Emulator For Ethernet Operation"*.

Parallel Connection
A parallel port is provided for high-speed code downloads. Connect a 25-pin "straight-thru" cable from the parallel port on the back of the emulator to the parallel port on the emulator. High-speed parallel code downloads will then be possible.

HMI-200 Series Emulators
The HMI-200 Series emulators can be connected via one of two methods:

Serial Connection
To use a RS-232 serial communications link, you must connect a serial cable from a serial port of the computer to the RS-232 serial port of the emulator. This should be a "straight-thru" or "modem" type cable. Speeds of up to 57.6K baud are supported on most Pentium computers with 16550 UARTs running Windows 95 or Windows NT.

Ethernet Connection *(Not Available On All HMI-200 Emulators)*
If you are using the optional Ethernet connection, make sure that your network cable is properly fastened to either the BNC or phone jack connections on the back of the emulator. Also, before the emulator can be accessed via Ethernet, it will need to be configured for Ethernet operation. For additional information on how to set up an emulator for Ethernet operation, see the HMI Application Note "*ACEE-A-100: Configuring An Emulator For Ethernet Operation*".

Parallel Connection *(Not Available On All HMI-200 Emulators)*
A parallel port is provided for high-speed code downloads. Connect a 25-pin "straight-thru" cable from the parallel port on the back of the emulator to the parallel port on the emulator. High-speed parallel code downloads will then be possible.

Background Mode Debuggers (BMDs)
BMDs can only be connected serially:

Serial Connection
To use a RS-232 serial communications link, you must connect a serial cable from a serial port of the computer to the RS-232 serial port of the BMD. This should be a "straight-thru" or "modem" type cable. Speeds of up to 57.6K baud are supported on most Pentium computers with 16550 UARTs running Windows 95 or Windows NT.

Simulators
Since the Simulators require no external hardware, no special hardware connections exist for this product.

HMI Hardware To Target System Connection

With the exception of the Background Mode Debuggers (BMDs), you do not need to have your HMI hardware connected to a target system for stand-alone operation. However, if you wish to emulate your target system, you must connect the HMI hardware to your target system.

For additional information on how to do this, see Chapter 4 of this manual.

If you experience problems interfacing to the target system or if you are just "not sure" about what you are doing, please do not hesitate to contact HMI Technical Support.

Starting SourceGate II

This section will describe how to start SourceGate II on your PC or UNIX system.

Starting The Windows Versions Of SourceGate II

The Windows versions of SourceGate II can be started from Windows in several ways. There are also environment variables and command line options that can be defined which influence the way SourceGate II will behave.

Windows 3.1x

To start SourceGate II from Program Manager, double-click on the SourceGate II icon located in the HMI program group. **This is the recommended method of starting SourceGate II .** You can also choose "File/Run" from the Program Manager menu bar and enter the path of the SG2.EXE file, along with any command line options, and then choose OK.

To start SourceGate II from File Manager, go to the directory where the SourceGate II files are located, highlight the SG2.EXE file, and choose OK.

Note: If no command line options are given when SourceGate II is invoked, the communications link will default to a standard RS-232 connection configured as COM1 and 9600 baud. For additional information on the available command line options, see the section ***Command Line Options***.

Windows 95 and Windows NT

Under Windows 95 and Windows NT, the easiest way to start SourceGate II is to press the Windows "Start" button, open "Programs", open "HMI", and click on the SourceGate II shortcut that was created during the installation process.

An alternate method of starting SourceGate II under Windows 95 and Windows NT is to click on the "Start" button, choose "Run...", enter the path of the SG2.EXE file, and then choose OK.

Note: If no command line options are given when SourceGate II is invoked, the communications link will default to a standard RS-232 connection configured as COM1 and 9600 baud. For additional information on the available command line options, see the section ***Command Line Options***.

Starting The Unix Versions Of SourceGate II

To start the Unix versions of SourceGate II, change to the directory where SourceGate II resides and enter the following:

```
sg2
```

This will start SourceGate II with the default interface type and baud rate parameters. If you wish to start SourceGate II using other optional parameters, they should be entered as follows:

```
sg2 <Command Line Options>
```

For additional information on the available command line options, see the section Command Line Options.

Command Line Options

Several command line options can be entered for the SourceGate II program. If you are using Windows 3.1x, add these parameters for the SourceGate II icon by clicking on it once. Then choose 'File/Properties' and enter the desired command line options on the command line after the .EXE filename.

If you are using Windows 95/NT, place the mouse cursor on the "Start" button and then press the right mouse button. Then choose "Open" and select the "Programs" folder by double-clicking on it. Choose the "HMI" folder (or whatever name was used during installation) by double-clicking on it. Place the mouse cursor on the shortcut that was created for the SourceGate program and then press the right mouse button. Select "Properties" and then enter the desired command line options in the "Target:" field after the .EXE filename.

If you are using a Sun or HP system, place the options in the command line string that invokes the software.

Be sure and leave a space between each option. After you are done adding options, choose "OK" to save the new properties (Windows only). The next time SourceGate II is started, the new options will take effect.

The available command line options are as follows:

-<Baud>
Select communications port baud rate. Allowable selections are: N(No serial port initialization performed), 110, 150, 300, 600, 1200, 2400, 4800, 9600 (default), 19200, 38400, and 57600.

Note that HMI BMDs only support 9600, 19200, or 38400 baud. In addition, the usage of 16550 UARTs is highly recommended for baud rates > 9600 for all systems. This command line option is not applicable for HMI CPU Simulators. Most Unix systems will not support baud rates > 38400.

-<Port>
Select communications port.

Windows -- SourceGate II automatically uses COM1 as it's default communications port, unless COM1 is not available, in which case it will use COM2. The -port option will override the default SourceGate II port selection and cause SourceGate II to use only the port named. Allowable options are: COM1, COM2, COM3, COM4, and AUX.

Unix -- SourceGate II automatically uses /dev/tty00 as it's default communications port. This option will override the default SourceGate II port selection and cause SourceGate II to use only the port named. Choose the appropriate name for your serial port with this option.

Note that this command line option is not applicable for HMI CPU Simulators.

-D<DownLoadPort>
Select parallel port for downloading.

Windows -- SourceGate II will default to PRN. Available options are LPT1, LPT2, and LPT3.

Unix -- SourceGate II will default to /dev/pp1. Choose the appropriate name for your parallel port with this option.

Note that this command line option is only applicable for HMI In-circuit emulators that are equipped with a parallel port.

-L<LineSymbol>
In order to enable source-level debugging, SourceGate II must know the relationship between the lines in the source file on disk, and the assembly language code being loaded into memory. This relationship is defined through the use of Line Number Symbols.

The default format for the line number symbols recognized by SourceGate II is "L_%u". This is a scanf-formatted string indicating that line number symbols are those symbols which consist of one capital 'L' character, followed by an underscore character and an unsigned decimal number. The unsigned decimal number is the line number. The value associated with the symbol is the starting address of the assembly-language instruction generated by that line of source code.

If, for example, your compiler produced line number symbols of the format LL_%u, then the -LLL_%u option should be used on the command line which initiates SourceGate II. In this example, the -L tells SourceGate II that the line number format is LL_%u. This command line switch allows source-level debugging using any compiler that produces line number symbols.

-E<SrcExt>
This switch defines the source code extension. The default extent is ".C". When source level or mixed mode debugging is enabled, SourceGate II will look for source modules with the defined extension.

-i<IP Address>
This option allows you to specify an IP address when an emulator equipped with the Ethernet option is used. The <IP Address> parameter should either be the actual IP address or the host name as listed in the "/windows/hosts" (Windows) or "/etc/hosts" (Unix) file.

If your "hosts" file contains the line:

```
192.9.200.201    hmi68000
```

then you could invoke SourceGate on this example Ethernet emulator by entering the following:

```
sg2 -ihmi68000
```

Note that this command line option is only available for those HMI emulators that have the Ethernet option installed.

-[NO]RTS
This option enables/disables the RTS/CTS handshaking between HMI systems and the computer system. If "-RTS" is used, the software will enable the RTS/CTS handshake. If "-NORTS" is used, the software will disable the RTS/CTS handshake. This hardware handshake may be needed if "COM PORT OVERRUN" messages appear at any time in a status window while transferring data.

Note that this command line option is not applicable for HMI CPU Simulators or for any HMI development tools being run on Unix systems.

-B<CommandFile>
This option defines a command (.CMD) file to be automatically executed upon entering SourceGate II.

<HexFile>
This option establishes the default program filename for RD and W commands.

Examples:

SG2.EXE	= Start SourceGate II with default options
SG2.EXE -19200	= Set 19200 Baud
SG2.EXE -COM2	= Use COM2
SG2.EXE -DLPT2	= Use LPT2 for parallel downloads
SG2.EXE -LLL_%u	= Set Line Number Symbol format to "LL_%u"
SG2.EXE -E68K	= Set extent default to ".68K"
SG2.EXE -i192.9.200.201	= Connect via Ethernet at the specified IP address
SG2.EXE -NORTS	= Disable RTS/CTS handshake
SG2.EXE -BTEST	= Run "TEST.CMD" at startup
SG2.EXE DEMO.OUT	= Define "DEMO.OUT" as default filename for RD and W

Environment Variables

Several environment variables are used by SourceGate II. These environment variables can be defined by using the DOS "SET" command or by using the Unix "setenv" command. These variables are defined as follows:

HMI_BAUD

Environment variable for the baud rate of the serial link between the computer and the emulator. Options are N(None), 110, 150, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, and 57600.

Note that HMI BMDs do not support 9600, 19200, or 38400 baud and most Unix systems do not support baud rates higher than 38400 baud. This option is not applicable for HMI CPU Simulators.

HMI_LINE

Environment variable for line number format. In order to enable source-level debugging, SourceGate II must know the relationship between the lines in the source file on disk, and the assembly language code being loaded into memory. This relationship is defined through the use of Line Number Symbols.

The default format for the line number symbols recognized by SourceGate II is "L_%u". This is a scanf-formatted string indicating that line number symbols are those symbols which consist of one capital 'L' character, followed by an underscore character and an unsigned decimal number. The unsigned decimal number is the line number. The value associated with the symbol is the starting address of the assembly-language instruction generated by that line of source code.

If, for example, your compiler produced line number symbols of the format LL_%u, then LL_%u option should be defined for this environment variable.

HMI_PORT

Environment variable for communications port being used to communicate between the computer and the emulator.

Windows -- SourceGate II automatically uses COM1 as its default communications port, unless COM1 is not available, in which case it will use COM2. This environment variable will override the default SourceGate II port selection and cause SourceGate II to use only the port named. Allowable options are: COM1, COM2, COM3, COM4, and AUX.

Unix -- SourceGate II automatically uses /dev/tty00 as it's default communications port. This option will override the default SourceGate II port selection and cause SourceGate II to use only the port named. Choose the appropriate name for your serial port with this option.

Note that this environment variable is not applicable for HMI CPU Simulators.

HMI_LPT
Environment variable for parallel download port to be used when downloading files from the computer to the emulator over the parallel link.

Windows -- SourceGate II will default to PRN. Available options are LPT1, LPT2, and LPT3.

Unix -- SourceGate II will default to /dev/pp1. Choose the appropriate name for your parallel port with this option.

Note that this environment variable is only applicable for HMI In-circuit emulators that are equipped with a parallel port.

HMI_RTS
Environment variable for default setting of the RTS/CTS handshaking between the computer and the HMI system. Options are Y (enable handshake) and N (disable handshake).

This hardware handshake may be needed if "COM PORT OVERRUN" messages appear at any time in a status window while transferring data.

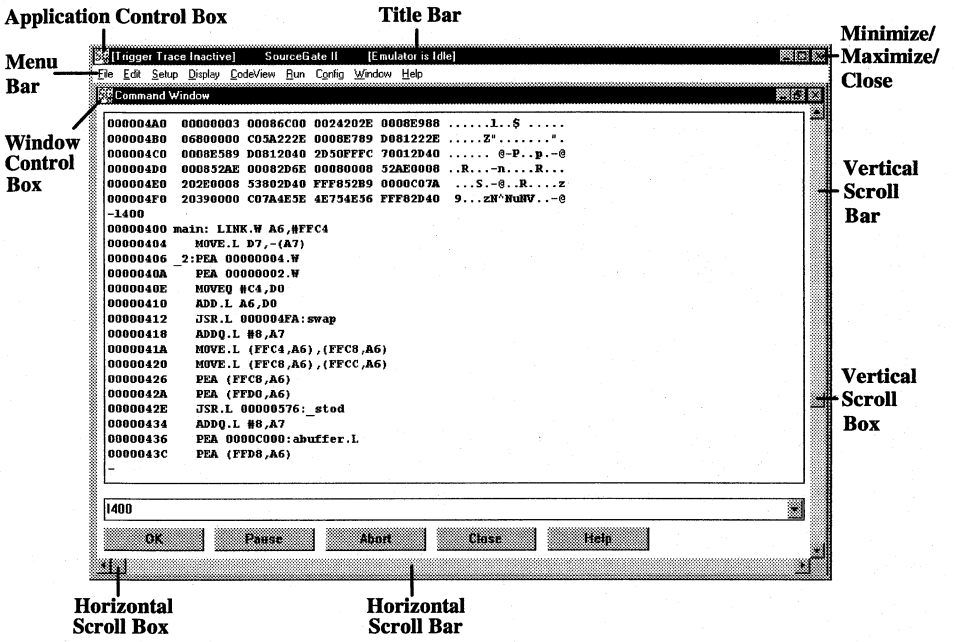
Note that this environment variable is not applicable for HMI CPU Simulators or for any HMI development tools being run on Unix systems.

HMI_STARTUP
Environment variable for setting the command file that gets executed when SourceGate II is invoked. This is handy for automating the loading of a file when SourceGate II is run.

2 General Overview

Becoming Familiar With The SourceGate II Window

Take a moment to review the sample SourceGate II window on this page. It is recommended that you become familiar with the various parts of the SourceGate II window so you can work quickly and efficiently.



An explanation of each of these window controls is given as follows:

Control Boxes

The Control Boxes display a menu with commands that allow you to restore, move, size, minimize, maximize, or close the application/active window. The application control menu box also allows you to switch between windows or open the control panel window.

Menu Bar

Lists the main menu items available in SourceGate II.

Title Bar
Displays the name of the application (SourceGate II) and shows the status of the emulator (Idle or Running) and the trace trigger (Active or Inactive).

Restore Button
Restores the SourceGate II application or window to the size and position it occupied before it was maximized or minimized.

Maximize Button
Enlarges the SourceGate II application or window to fill the entire screen.

Minimize Button
Reduces the SourceGate II application or window.

Scroll Bars (Horizontal and Vertical)
Controls the portion of window text that appears on the screen.

Scroll Boxes (Horizontal and Vertical)
Indicates where the screen display is located compared to the full width of the window.

SourceGate II Menu Bar

The menu bar for SourceGate II provides you a way to pull up the main menu items that can be used with SourceGate II. An explanation of each of these menu items follows. Note that the menu bar choices might change depending on the current active window. The following menu choices are available when the main SourceGate II window is active. **If a menu item is grayed out, it is either not available on your particular system or the menu item has not been implemented in this version of SourceGate II.**

The 'File' Menu

The main function of the 'File' menu is getting information into and out of SourceGate II. The following is a brief explanation of the options available in this menu:

Download Object File...
Download an object file into memory.

Upload Object File...
Upload memory into an object file located on the host that can be used to burn PROMs.

Execute...
Execute a pre-recorded SourceGate II command file.

Record...
Record a command file

Log...
Begin logging all Command window keyboard input and screen output to an ASCII file that can be reviewed at a later date. This enables you to obtain a "log" file of what you have done during the session.

Print
Print the contents of the active window to a file or a printer. **This menu item will be grayed out if the active window does not support printing operations.**

Printer Setup...
Set up printer parameters.

Exit SourceGate...
Exit the SourceGate II program.

The 'Edit' Menu
The 'Edit' menu items are used in cut and paste operations. This menu item is not implemented in this version of SourceGate II and is grayed out.

The 'Setup' Menu
The 'Setup' menu is used to set up events, breakpoints, trigger points, and other system specific operations. The information shown in this window varies depending on the HMI system that you are using. For additional information for your particular system, see Chapter 4 of this manual.

The 'Display' Menu
The 'Display' menu allows the user to choose certain windows for display. The following is a brief explanation of the options available in this menu:

Command Window
Opens the Command window which allows you to enter any system command and display the results of that command.

Performance Analysis (Requires PAC - Not Available On BMDs)
Opens the Performance Analysis window which allows you to perform detailed timing analysis of your code.

Trace (Not Available On BMDs)
Opens the Trace window which allows you to actually see what instructions that the system is executing.

Watch
Opens the Watch window which allows you to change and display variables of interest.

Watch In Real Time (Not Available On BMDs and Certain 200 Series Emulators)
Opens the Watch In Real Time window which allows you to change and display variables of interest while running in real-time.

Registers
Opens the Register window which allows you to display and change the processor's register values.

Memory
Opens the Memory window which allows you to display and change memory locations.

Stack Trace
Opens the Stack Trace window which shows the functions and variables that have been pushed on to the stack.

Message Window
Opens the Message window which allows you to view any messages sent by the system to SourceGate II.

C++ Class Browser
Opens the C++ Class Browser window which allows you to view all lines of inheritance and C++ type information.

pSOS Data
Opens the pSOS Data window which allows you to view several different types of pSOS data.

The 'CodeView' Menu
The main function of the 'CodeView' menu is to provide an interface that deals strictly with source files. The CodeView interface should be treated as a separate system within SourceGate II. The following is a brief explanation of the options available in this menu:

Modules...
Opens the 'Module Selection' window which allows you to view and select a source module to work with.

Symbols...
Opens the 'Symbol Selection' window which allows you to view and select symbols for a particular module.

List BPs
Lists all set CodeView window breakpoints.

View PC
Gives focus to the CodeView window that contains the program counter. Opens a new CodeView window if no open windows contain the current PC value.

VASM
Opens an assembly window beginning at the address defined by the user.

PC Tracking Module
While stepping, this window will switch to the CodeView window that contains the program counter, opening new windows if needed.

Cursor Tracking Assembly
Invokes an assembly window which will remain on the screen and will track the source window that currently has the focus by highlighting the assembly lines that correspond to the source line highlighted in the active module. If this window is activated when no source modules are open, it will show assembly code beginning at the current program counter location.

Start
Begins emulation from the current program counter location. This item will be grayed out if emulation is in progress.

It is important to note that starting emulation using the 'Go' command from either the Command window, Emulation Status window, or 'Run/Go' menu item will not use the breakpoints defined in the CodeView interface but will use either the breakpoints defined as part of the 'Go' command or the event based breakpoints.

Stop
Stops emulation and update all window values with the latest information. This item will be grayed out if emulation is not in progress.

Reset
Resets the system processor. This is the same procedure as toggling the RESET' line on the processor or entering the "RS" command.

Step In
Steps into any functions that are not contained in the active window where stepping is taking place, opening up new windows as stepping progresses.

Step Over
Steps over any functions that are not contained in the active window where stepping is taking place. The "stepped over" functions will actually be executed, you will just not see the actual stepping of these instructions. Note that stepping over a long routine may take a few moments to complete.

The 'Run' Menu

The 'Run' menu contains commands that deal with starting and stopping emulation. The following is a brief explanation of the options available in this menu:

Emulator Status
Opens the Emulation Status window which allows you to monitor the status of the system and to control emulation.

Go From/To...
Opens the Go From/To window which allows you to start and stop emulation at a certain point.

Go
Begins emulation from the current program counter location. This item will be grayed out if emulation is in progress.

It is important to note that starting emulation using the 'Go' command from either the Command window, Emulation Status window, or 'Run/Go' menu item will not use the breakpoints defined in the CodeView interface but will use either the breakpoints defined as part of the 'Go' command or the event based breakpoints.

Stop
Stops emulation and updates all window values with the latest information. This item will be grayed out if emulation is not in progress.

Reset
Resets the system processor. This is the same procedure as toggling the RESET' line on the processor or entering the "RS" command.

The 'Config' Menu

The 'Config' menu deals with setting up system options. The following is a brief explanation of the options available in this menu:

Preferences
Allows you to customize how the CodeView interface looks and behaves. Also allows you to define how object files are loaded.

Register Map
Defines the values to be used for register variables.

Compiler Interface...
Defines the relationship between SourceGate II and the compiler being used.

Emulator Memory Map... (Not Available On BMDs and Simulators)
Configures the system's memory map and other signal parameters.

Simulator (Simulators Only)
Configures the special features of the Simulator.

Serial Interface... (Not Available On BMDs and Simulators)
Views the RS-232 serial link connection parameters between the computer and the system.

Restore Memory Map... (Not Available On BMDs and Simulators)
Restores a previously saved system memory map file.

Save Memory Map As... (Not Available On BMDs and Simulators)
Saves the current system memory map configuration to a file.

Factory Configuration... (Not Available On BMDs and Simulators)
Returns all system parameters to their factory default values. This is equivalent to performing the 'CONFIG' command from the Command window.

Reset Serial Port (Not Available On BMDs and Simulators)
Resets the serial port between the computer and system and re-initializes the connection.

The 'Trace' Menu (Not Available For BMDs)

The 'Trace' menu item is shown when the Trace window has been activated via the 'Display/Trace' menu item. The following is a brief explanation of the options available in this menu. Note that not all of these options are available for all systems. Trace features for your particular HMI system are discussed in more detail in Chapter 4 of this manual.

Freeze
Takes a "snapshot" of what is being executed by the CPU and shows this information in the Trace buffer.

Resume
Resumes normal trace operation after a "Freeze" command has been issued.

Break Buffer
Shows Break buffer information in the Trace window.

Trigger Buffer
Shows Trigger buffer information in the Trace window.

View
Allows you to define the Trace window display format (Disassembly, State, etc.).

Timing
If PAC is available, shows a Relative or a Delta timestamp in the Trace window.

Goto Top
Go to the top of the trace buffer.

Goto Bottom
Go to the bottom of the trace buffer.

Find Trigger
Finds the Trace Trigger point in the trace buffer if one is available.

Ports
Shows port values for certain emulators.

Filter
Filters irrelevant data out of the trace buffer on certain emulators.

The 'PAC' Menu *(Requires PAC - Not Available On BMDs)*

The 'PAC' menu item is shown when the Performance Analysis window has been activated via the 'Display/Performance Analysis' menu item. The following is a brief explanation of the options available in this menu:

Profile Setup...
Defines the modules of interest to be analyzed and other PAC setup options.

Profile Data...
Shows numerical data results of Performance Analysis runs.

Profile Graph...
Shows graphical histogram results of Performance Analysis runs.

Coverage
Shows hexadecimal addresses that were covered during the run.

Extended Coverage
Shows the actual assembly language instructions that were covered during the run.

Extended Source
Shows the actual high-level language source instructions that were covered during the run.

Restore PAC Setup...
Restores a previously saved PAC setup.

Save PAC Setup...
Saves the current PAC setup to a file.

Close PAC
Closes the PAC window.

The 'Window' Menu

The 'Window' menu allows you to manipulate the SourceGate II windows. The following is a brief explanation of the options available in this menu.

Next
Open the next window in the open window list of previously opened windows.

Previous
Make the window that was active prior to the current window to be the active window.

Cascade
Cascades all open windows.

Tile Horizontal
Tiles all open windows horizontally. **Note: This option is not supported in this release of SourceGate II and is grayed out.**

Tile Vertical
Tiles all open windows vertically. **Note: This option is not supported in this release of SourceGate II and is grayed out.**

Close
Close the open window that currently has focus.

Close All
Close all open windows.

Open Window List
Not an actual option, this list will show all windows that are currently open. You can make a window in this list to be the active window by clicking on it. If no windows are open, "No Windows" will be shown.

The 'Help' Menu

The 'Help' menu provides help on using SourceGate II. The following is a brief explanation of the options available in this menu:

On Context
Provides help on the current active window. This is the same as 'On Window'. Future versions of SourceGate II may use this option to provide help on particular fields instead of at the window level.

On Window
Provides help on the current active window. This is the same as 'On Context'.

Keyboard
Provides a listing of the commands that can be executed from the keyboard.

Index
Shows an index of the topics available in the SourceGate II Help System.

How Do I...
Provides help for some of the more commonly asked questions.

Help On Help
Provides help on how to use the help system.

Search
Allows you to search for a particular topic by using keywords.

About
Shows information about the version of the help files.

Window Manipulation

Several windows can be shown at the same time. However, you may wish one window to "have focus" or be the top-most window. The following sections describe how to work with multiple windows.

Changing To The Next Window

Once several windows have been opened, you may choose the 'Window' menu item. You will then be shown a listing of the windows that are open. A check mark will be next to the active window. To make the window listed below the checked one to be the active window, choose ***Next*** or press the F6 key.

Changing To The Previous Window

Once several windows have been opened, you may choose the 'Window' menu item. You will then be shown a listing of the windows that are open. A check mark will be next to the active window. To make the window listed above the checked one to be the active window, choose ***Previous*** or press the "Shift+F6" key combination.

Choosing From The Open Window List

Once several windows have been opened, you may choose the 'Window' menu item. You will then be shown a listing of the windows that are open. A check mark will be next to the active window. To make another window in the list to be the active window, simply click on it.

Cascading All Open Windows

To cascade all open windows, choose ***Cascade*** or press the "Shift+F5" key combination.

Closing The Current Active Window

To close the current active window, choose ***Close*** or press the "Ctrl+F4" key combination.

Closing All Open Windows

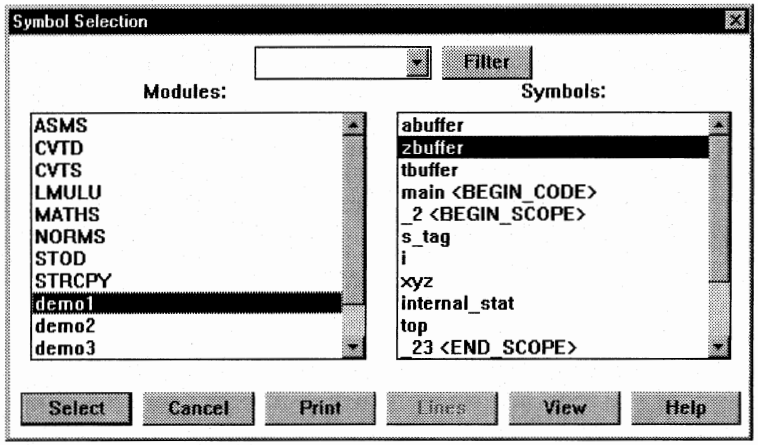
To close all windows that are currently open, choose ***Close All*** or press the "Ctrl+Z" key combination.

Defining Your Printer

You can define the printer you wish to use with SourceGate II by choosing 'File/Printer Setup...'. You will be shown the current printer and you can then change to another printer, if you wish, provided you have another printer available.

Working With Symbols

While you are in any field in SourceGate II from which symbols can be entered, pressing the ***right mouse button*** (or the "Ctrl+Y" key combination) under Windows (middle mouse button under Unix) opens the Symbol Selection window which allows you to select a symbol for that field. An example Symbol Selection window is shown below:



The Symbol Selection window can also be invoked from the main menu by choosing "CodeView/Symbols...". The symbol information shown will be for the highlighted module. Further, either line number information or symbols for that module will be shown. Some windows will only allow line number information to be entered. In this case, the "Symbol" button will be grayed out. Some windows will only allow symbols to be entered. In this case, the "Lines" button will be grayed out. Still other windows will allow either set of information to be entered by toggling the "Line/Symbol" button. The button shown will be the opposite of the currently displayed symbol format so that the user can toggle to the other mode if so desired.

The fields shown in this window are defined as follows:

Symbol Selection Window 'Modules' Field

The 'Modules' field shows the modules that have symbol information for them loaded into the symbol table.

Symbol Selection Window 'Symbols' Field

The 'Symbols' field shows the symbols (or line numbers depending on mode) that are available in the symbol table for the selected module.

Symbol Selection Window Buttons

The following buttons are available in the Symbol Selection window:

Select

The **Select** button will select the highlighted symbol. This allows the symbol to be used in any field that accepts hexadecimal data, placing the actual value of the symbol in this field.

Note: The Select button is not available when the Symbol Selection window is invoked from the "CodeView/Symbols..." menu item or via the "Ctrl+Y" key combination. When invoked via these methods, the Symbol Selection window is presented as an "information only" window and the Select button is not available.

Filter

The **Filter** button allows the user to filter the symbols that are displayed. If the user enters a string in the filter field and then presses the **Filter** button, only symbols for the selected module that begins with the entered string are displayed. The filter will accept wildcard characters (e.g. "*buf*" or "a*?l" are valid). The **Scroll Box** button can be used to view and select previously entered filter strings. Filter strings are saved in a history list and the current filter string is saved between invocations of the Symbol Selection window.

Close
The **Close** button closes the Symbol Selection window.

Cancel
The **Cancel** button will cancel the symbol selection process and will close the Symbol Selection window.

Print
The **Print** button will allow the symbol information to be output to a file or a printer.

Lines
The **Lines** button will show the line number symbols associated with the highlighted module. This button may be grayed out when the Symbol Selection window is invoked from certain windows and it will be replaced by the **Symbols** button when line number symbols are being shown.

Symbols
The **Symbols** button will show the symbols associated with the highlighted module. This button may be grayed out when the Symbol Selection window is invoked from certain windows and it will be replaced by the **Lines** button when symbols are being shown.

View
The **View** button allows the user to view the characteristics of the selected module, symbol, or line number symbol. If the user first clicks on a module and then presses the **View** button, the View Module window will be shown. If the user first clicks on a symbol and then presses the **View** button, the View Symbol window will be shown. If the user first clicks on a line number symbol and then presses the **View** button, the View Line window will be shown.

Help
The **Help** button will provide help for this window.

Scroll Box
The **Scroll Box** button will allow the user to scroll through previously entered filter strings.

Using Help

SourceGate II provides on-line Help for every window. You can resize, move, or cascade the Help window and the current active window so that you can keep both of them displayed.

There are several ways to obtain on-line Help:

- Context sensitive Help
- Help for the current active window
- Help menu

Currently, 'context sensitive help' and 'help for the current active window' perform the same function. Help is always provided for the current active window if either of these options is chosen. You can also always use the accelerator key F1 to obtain help on the current active window.

Context Sensitive Help

Choose 'Help/On Context'. SourceGate II will then provide Help for the active window (same as ***Active Window Help*** for this release).

Active Window Help

Choose 'Help/On Window'. SourceGate II will then provide Help for the active window (same as ***Context Sensitive Help*** for this release).

Help Menu

The Help menu will provide help on all areas. To invoke the Help menu, choose 'Help'. The following is a description of each available menu item:

On Context

Provides help on the current active window (same as ***On Window*** for this release).

On Window

Provides help for the current active window (same as ***Active Window Help*** for this release).

Keyboard

Provides keyboard shortcuts (accelerator keys) for SourceGate II functions and commands.

Index

Displays a list of available help topics from which you can gain access to other parts of the SourceGate II Help System.

How Do I...

Displays a list of common SourceGate II questions and provides answers to each.

Help On Help

Provides help on how to use the help system. The information shown here will be provided by the help viewer and is a general overview of how help works under the operating system that you are using.

Search

Invokes the Help system's Search engine allowing you to search for a particular topic or keyword. Once the topic or keyword is found, a list of topics related to that topic or keyword is shown. You can then choose from one of these topics for additional information on the topic.

About

Provides information about the version of the Help files. The following is an example of what you may see when you choose this menu item:

SourceGate II 68060 Help System
File Edit Bookmark Options Help
Contents Search Back Print
Version Information
SourceGate II
SourceGate II(tm) (Use VER command to see SGII Version)
HMI-200/SPS-2000 Series Emulator
Main Help File - Revised 02/13/97
68060 Help File - Revised 02/09/97
Copyright (c) 1997 by:
HMI
Huntsville Microsystems, Inc.
(205) 881-6005 (Voice)
(205) 882-6701 (FAX)
(205) 881-7395 (BBS) Up to 14400 Baud, 8-N-1
Technical Support Internet Address: support@hmi.com
Sales Internet Address: sales@hmi.com
World Wide Web Address: http://www.hmi.com/
FTP Site Address: ftp://ftp.hmi.com/

Notice that the version of SourceGate II is not shown in this window. To find out what version of SourceGate II that you are using, use the "VER" command.

Exiting SourceGate II

To exit SourceGate II, you can do one of the following:

- Choose 'File/Exit SourceGate'
- Press Ctrl+Q
- Double-Click on the SourceGate II window application control box

A window will then appear confirming that you really do wish to quit. Choosing 'Exit' will exit SourceGate II. All active window sizes and positions will be saved so that they will appear the same the next time SourceGate II is started

Accelerator Keys

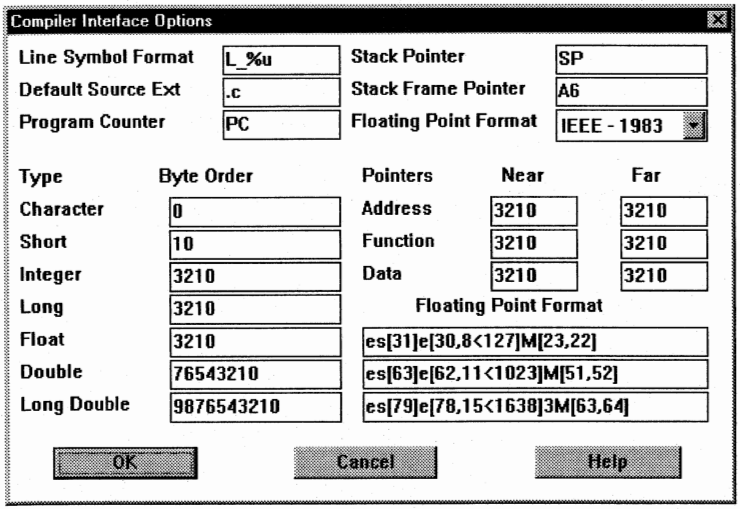
You can use accelerator keys, shown beside some commands, to select a command. For example, Ctrl+Q will exit SourceGate II. The following is a listing of all available accelerator key combinations. Note that not all accelerators are available for all systems.

This key...	...does this
Alt+C	Show CodeView Menu Options
Alt+D	Show Display Menu Options
Alt+E	Show Edit Menu Options
Alt+F	Show File Menu Options
Alt+H	Show Help Menu Options
Alt+O	Show Config Menu Options
Alt+P	Show PAC Menu Options (PAC Must Be Active)
Alt+R	Show Run Menu Options
Alt+S	Show Setup Menu Options
Alt+T	Show Trace Menu Options (Trace Must Be Active)
Alt+W	Show Windows Menu Options
Alt+F4	Exit SourceGate II
Ctrl+A	Open Cursor Tracking Assembly Window
Ctrl+B	Define Breakpoint (200 Series)/ Define Sequence Definition (SPS-2000 Series)
Ctrl+C	Open Command Window
Ctrl+D	Download Object File
Ctrl+E	Open Emulator Status Window
Ctrl+F	Find
Ctrl+G	Go (Begin Emulation - Use Hardware Breakpoints)
Ctrl+H	View PC

This key...	...does this
Ctrl+I	Define Interval Timer (200 Series)/ Define Timers (SPS-2000 Series)
Ctrl+J	Define Trace Trigger (200 Series)/ Define Sequence Program (SPS-2000 Series)
Ctrl+K	Open Stack Trace Window
Ctrl+L	List CodeView BPs
Ctrl+M	Open Memory Window
Ctrl+N	Activate Performance Analysis System
Ctrl+O	Display Modules
Ctrl+P	Open PC Tracking Module Window
Ctrl+Q	Exit SourceGate II
Ctrl+R	Open Register Window
Ctrl+S	Save To File
Ctrl+T	Open Trace Window
Ctrl+U	Upload Object File
Ctrl+V	Open VASM Window
Ctrl+W	Open Watch Window
Ctrl+X	Execute Command File
Ctrl+Y	View Symbols
Ctrl+Z	Close All Windows
Ctrl+F1	Search The Help
Ctrl+F4	Close Active Window
Shift+F5	Cascade Open Windows
Shift+F6	Previous Open Window
F1	Invoke Help For Active Window
F3	Find Next
F5	Start (Begin Emulation - Use CodeView Breakpoints)
F6	Next Open Window
F7	CodeView Step In
F8	CodeView Step Over
F9	Stop Emulation
F12	Reset CPU

Setting Compiler Interface Options

The Compiler Interface Options window allows the user to define compiler and processor specific parameters such as the line number symbols generated by the compiler, the default source extent used by SourceGate, the floating point format, the byte order for all variable types and pointers, and the registers to be used for the stack and stack frame pointers. This window is invoked via the "Config/Compiler Interface..." menu item. An example Compiler Interface Options window is shown below:



Although these parameters can be changed by highlighting the desired section and entering changes, the default values are adequate in most instances and should not be changed unless absolutely necessary for proper operation.

The parameters defined in this window will be saved in a file called "<processor>.ini" (where <processor> is the name of the processor type being used; e.g. 68000.ini) which will be read and used as the default values for SourceGate until they are changed again by the user.

An overview of each of these fields is given in the sections that follow:

Line Symbol Format Field

This field determines the format of the line number records that SourceGate will use for source-level debugging. Line number records tell debuggers the lines of source that generate assembly code and where this assembly code begins in memory. The default format for SourceGate is L_%u where %u is an unsigned integer (e.g. L_10, L_23, etc.). If this default setting does not produce satisfactory results, consult your compiler documentation for the correct setting.

Default Source Ext Field

This field determines the extension of the user's source code files. SourceGate needs the proper extension as the actual source files will be opened during a CodeView session.

Program Counter Field

This field determines the register to be used as the program counter. Consult your compiler documentation for the proper register to use in this field.

Stack Pointer Field

This field determines the register to be used as the stack pointer. Consult your compiler documentation for the proper register to use in this field.

Stack Frame Pointer Field

This field determines the register to be used as the stack frame pointer. Consult your compiler documentation for the proper register to use in this field.

Floating Point Format Field

The Floating Point Format field will display a selection window that will allow IEEE-1983, IEEE- 1985, or the "Defined Fmt" to be used when dealing with floating point numbers. The "Defined Fmt" is the format that is entered in the lower "Floating Point Format" field by the user. The "Scroll Box" button can be used to display these different options.

Type/Byte Order Fields

These fields determine the byte order in memory for the defined variable type.

Pointers/Near/Far Fields

These fields determine the byte order in memory for near and far pointers.

Floating Point Format Fields

These fields determine the floating point format to be used if "Defined Fmt" is selected in the preceding "Floating Point Format" field.

Compiler Interface Options Window Buttons

The buttons in the Compiler Interface Options window allow the user to perform certain actions. An explanation of all available buttons is given as follows:

OK

The **OK** button will accept the entered data for this window and will store this information in the ".INI" file for future use.

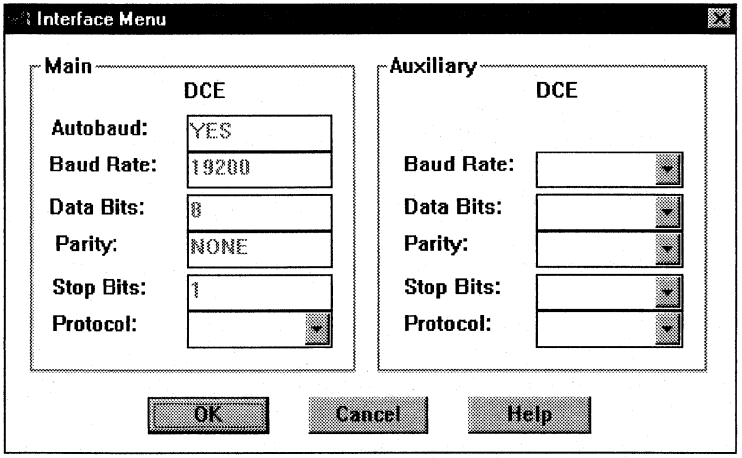
Cancel
The **Cancel** button will cancel the current operation and close the window. None of the changes that were made will be saved to the ".INI" file.

Help
The **Help** button will invoke help for this window.

Scroll Box
The **Scroll Box** button will allow the user to scroll through the available choices for the floating point format.

Viewing Serial Interface Parameters

The Interface window shows the current configuration of the Main and Auxiliary RS232 (if installed) ports. An example Interface window is shown below:



The attributes of the main port cannot be changed from this menu (with the exception of the handshaking protocol), but are shown to reflect the switch setting for switch S1 or, for autobaud mode, the software switches entered when starting the program.

The configuration for the auxiliary port is configurable from this menu. The user can use the scroll box buttons to pick from the available choices for the baud rate, number of data bits, parity, stop bits, and handshake protocol. These settings should match the device being connected to the auxiliary port. **Note that only certain 200 Series emulators have an Auxiliary Serial port available.**

Interface Window Buttons

The buttons in the Interface window allow the user to perform certain actions. An explanation of all available buttons is given as follows:

OK

The **OK** button will accept the entered data for this window.

Cancel

The **Cancel** button will cancel the current operation and close the window. None of the changes that were made will be saved.

Help

The **Help** button will invoke help for this window.

Scroll Box

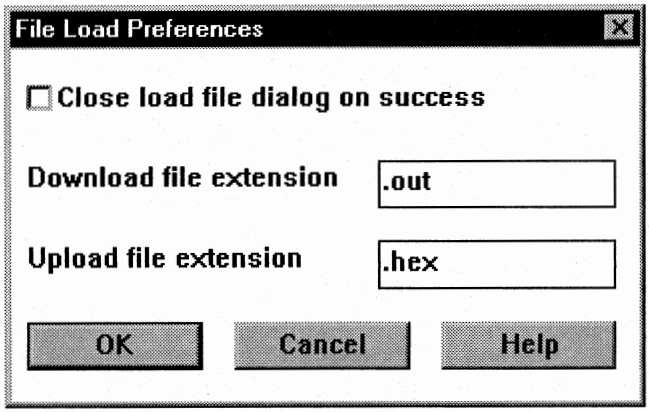
The **Scroll Box** button will allow the user to scroll through the available choices for each parameter.

Configuring User-Defined Preferences

The user can define preferences for the CodeView windows and file loading operations by several methods. These are outlined in the following sections:

File Loading Preferences

To change the file load preferences, select the "Config/Preferences/File Load..." menu item. This will open the "File Load Preferences" window:



Each of the fields shown in this window are outlined in the following sections.

Close Load File Dialog On Success
This option, when checked, will close the file load status window upon successful completion of the file load process. If this option is not checked, the file load status window will remain on the screen until the "Close" button is pressed.

Download File Extension
This field determines the file extent that is to be used when downloading a file using the 'File/Download Object File' command.

Upload File Extension
This field determines the file extent that is to be used when uploading a file using the 'File/Upload Object File' command.

File Load Preferences Window Buttons
The following buttons are available in this window:

OK
The **OK** button will accept the preferences entered.

Cancel
The **Cancel** button will cancel the entering of the preferences and will close the Preferences window.

Help
The **Help** button will invoke help for this window.

CodeView Window Preferences
Extensive user preferences are available for the CodeView windows. For additional information on setting CodeView window preferences, see *Setting CodeView Window Preferences* in Chapter 5 of this manual.

INI File Settings
Each HMI system has an associated "INI" file (68332.INI, MPC505.INI, etc.) which defines certain preferences when SourceGate II is started. These settings must be installed in the applicable INI file before SourceGate II is started in order for them to take effect. For the latest information on available INI file settings, reference the on-line help or contact HMI Technical Support.

Working With Object Files

The heart of any emulation system is the code that is loaded into memory. In the case of a working target system, this code usually will reside in a programmable device such as a PROM. However, when a system is under development, burning new PROMs every time the software changes can become a time-consuming chore. With SourceGate II and a HMI development system, this unpleasant task is a thing of the past until you have the final working version of your code developed.

SourceGate II can read a variety of object file formats directly into memory without requiring a conversion process. The formats supported include:

- Motorola S-Record *
- Intel Hex *
- Tekhex *
- HMI Binary *
- Common Object File Format (COFF)
- Software Development Systems (SDS)
- IAR Systems
- IEEE-695
- ELF/DWARF
- Extended Tekhex
- Microtec Research Absolute
- Hitachi

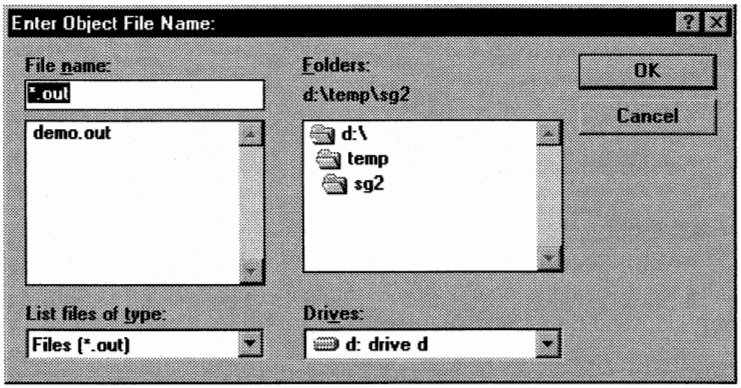
  * - No symbol information (including line numbers) supported in this format

Most compiler/assembler manufacturers provide linkers that can produce one of the above formats. Other formats can be converted to a format that SourceGate II can read through the use of a Converters Disk supplied free of charge by HMI.

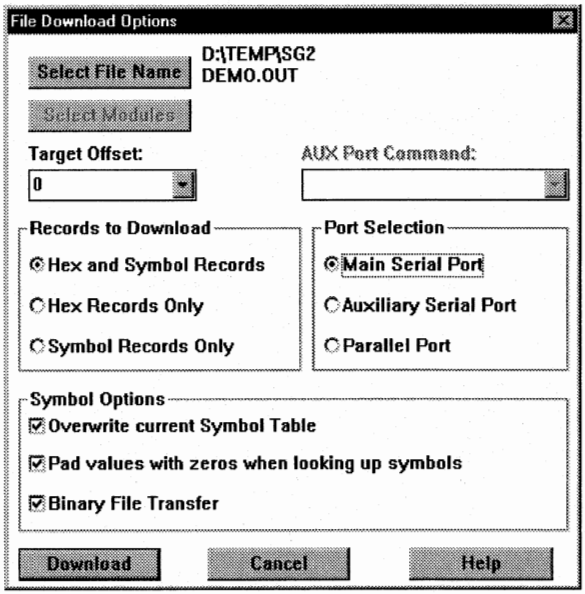
If you have further questions about file format support from a particular compiler manufacturer, call your compiler manufacturer and ask if they provide full debugging support for object files output in one of the above file formats.

Downloading An Object File

To download an object file into system memory, choose "File/Download Object File..." or press the "Ctrl+D" key combination. The user will be prompted to enter the name of the file to be downloaded:



Once the filename is selected, the File Download Options window will be displayed. This window provides additional options for file downloading:



After selecting from the options shown in this window, pressing the **Download** button will download the file into the system memory. If an emulator is being used, the file contents will either go into emulation or target memory, depending on the memory mapping. If a simulator is being used, the file contents will go into the "simulated" memory that resides in the host computer. If a BMD is being used, the file contents will go into the target system's memory via the debug port on the BMD.

File Download Options Window Fields

This window allows you to define the file to be downloaded using several options. An overview of each of these options is as follows:

'Target Offset' Field

This field allows you to enter an offset value. This number will shift the load address of each record in the file being loaded. For example, if you enter a target offset value of 8, then a record that is specified as loading at 400H will instead load at 408H.

'AUX Port Command' Field

This field will allow you to enter a command to be sent out the auxiliary port before the read begins. This command may be a string to a mainframe that directs the mainframe to begin sending the file, for example.

NOTE: This field is only changeable if you have "Auxiliary Serial Port" selected in the Port Selection field and the Auxiliary port is available on your system.

'Records To Download' Fields

These fields allows you to define what records get loaded. The options are as follows:

Hex and Symbol Records - All records in the file will be read.

Hex Records Only - Only the hex code portion of the file will be read. All symbol information is ignored.

Symbol Records Only - Only the symbol portion of the file will be read. All hex code information is ignored.

'Port Selection' Fields
These fields allow you to define the which port is used for the download. The options are as follows:

Main Serial Port - The file will be downloaded using the main serial port.

Auxiliary Serial Port - The file will be downloaded using the aux. serial port.

Parallel Port - The file will be downloaded using the parallel port.

'Symbol Options' Fields
If symbols are being downloaded, these options influence their transfer. The options are as follows:

Overwrite Current Symbol Table - Any current symbol table located in memory will be overwritten with the new symbol table information. Otherwise, the new symbol table will be added to the old information.

Pad Values With Zeros When Looking Up Symbols - All symbols read will have zeros padded at the beginning of their address. For example, a symbol with the value of 400H will be stored as 00000400H in memory. This feature can help qualify addresses in the trace and assembly windows which eliminates erroneous listings of symbols in these windows.

Binary File Transfer - All files will be converted to binary before transmitting. This will speed up the file transfer over transmitting in ASCII.

File Download Options Window Buttons
The following buttons are available in this window:

Select File Name
This button allows you to specify the path and file to be downloaded.

Select Modules
This button allows you to specify the individual modules to be loaded from a file. This can help speed up file transfers due to the loading of select module symbol information.

NOTE: This is a future enhancement not supported in this release of SourceGate.

Download
This button begins the file downloading process. This button will be grayed out if no filename has been selected.

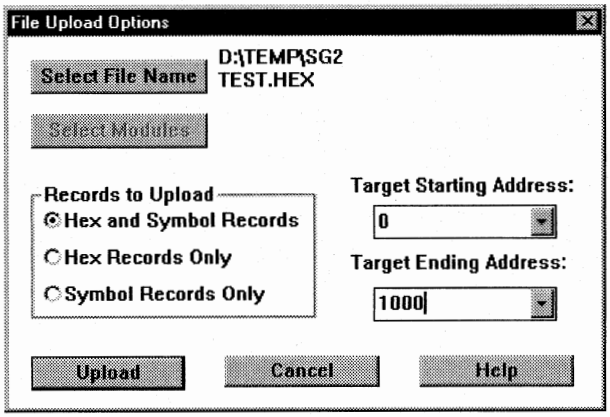
Cancel
This button cancels the file loading process.

Help
This button invokes help for this window.

Scroll Box
The **Scroll Box** button allows you to choose from previously entered target offset values or AUX port commands.

Uploading An Object File

After you have a working version of your software, you probably will want to save it to a file so that you can burn a PROM or FLASH module and then test it in your system. To do this, choose 'File/Upload Object File...' or press the "Ctrl+U" key combination. You will then be prompted to enter a filename. After a filename has been selected, you will be asked to enter other parameters in the File Upload Options window as shown below:



File Upload Options Window Fields

A description of the other fields available in this window follows:

Records To Dump
This section consists of three options which determine the information that is saved as part of the dumped file. The following is an overview of each of these options:

> **Hex and Symbol Records** - Dump both hexadecimal code and any symbol information to the designated file.
>
> **Hex Records Only** - Dump only the hexadecimal code to the designated file. Suppress the dump of any symbol information.
>
> **Symbol Records Only** - Dump only the symbol information to the designated file. Suppress the dump of any hexadecimal code.

Target Starting Address
This field defines the beginning address of the memory to be uploaded.

Target Ending Address
This field defines the ending address of the memory to be uploaded.

File Upload Options Window Buttons
A description of the functions of the buttons available in this window are as follows:

Select File Name
The **Select File Name** button will prompt you for the name of the file to be saved.

Select Modules
The **Select Modules** button allows you to specify from which modules you want symbol information uploaded into the output file.

NOTE: This is a future enhancement not supported in this release of SourceGate.

Upload
The **Upload** button begins the upload process to the selected file.

Cancel
The **Cancel** button cancels the upload operation and closes this window.

Help
The **Help** button invokes help for this window.

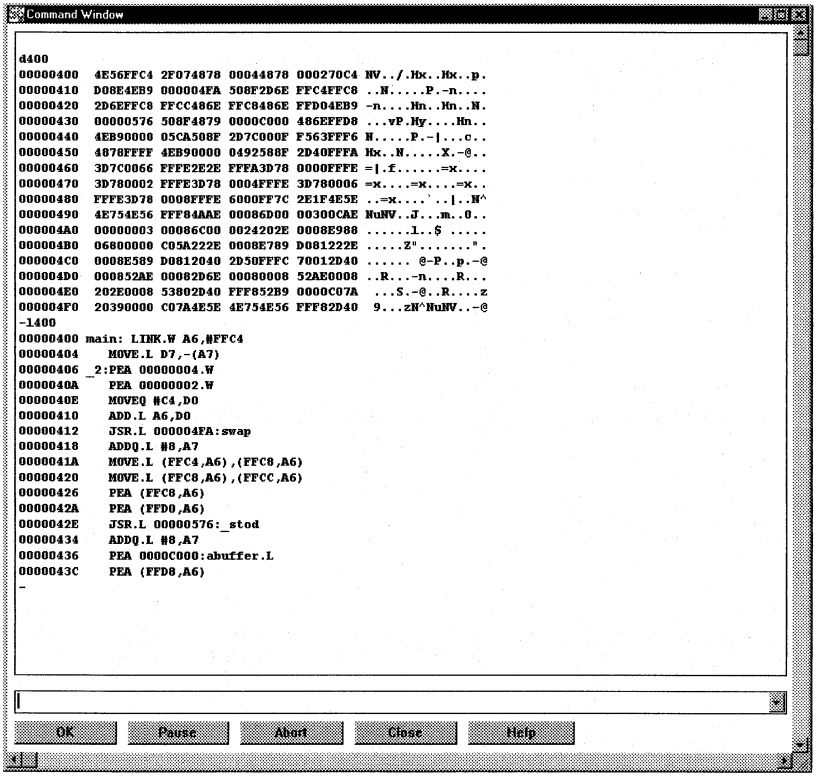
Scroll Box
The **Scroll Box** button allows you to select from previously entered values.

3 Using SourceGate Windows

SourceGate II has a number of windows that will prove invaluable to you during your development effort. An overview of each these windows is given in the sections that follow:

Using The Command Window

The Command window is used to enter commands to the system. While many of the Command window commands can be performed through other windows, the Command window provides a central location where all system commands can be performed. The Command window has the following format:



An overview of the operation of the Command window follows.

Opening The Command Window

To open the Command window, choose 'Display/Command Window'. An alternate method of opening the Command window is to use the accelerator key combination "Ctrl+C".

Entering Command Window Commands

All commands are entered in the command entry field located in the Command Window. Once a command is entered, the "Enter" key or "OK" button will initiate execution of the command. Previously entered commands can be recalled by pressing the "Scroll Box" button located next to the command entry field. A previously entered command can be highlighted using the mouse or the arrow keys. The highlighted command can be then be executed by pressing the "Enter" key or "OK" button.

In the Command Window, the command abbreviation and the first parameter are not required to be separated, but may be separated by one or more spaces. If multiple parameter fields are used, they must be separated by a recognized delimiter. Allowed delimiters include a comma, a single space, or multiple spaces. If a comma is used as the delimiter, it may be preceded and followed by one or more spaces. This allows you some flexibility in the entering of commands.

Available Command Window Commands

The commands available while in the Command window depend upon the system being used. For a detailed listing of the commands available for your particular system, see ***Available Command Window Commands*** in Chapter 4 of this manual.

Command Window Buttons

A description of the functions of the buttons available in this window are as follows:

OK

The **OK** button always executes the command displayed in the command entry field.

Pause

The **Pause** button stops the information from scrolling on the screen. After the **Pause** button is pressed, the button will display **Resume**. Pressing the **Resume** button will allow the scrolling to continue.

Abort

The **Abort** button will abort the command that is currently executing.

Close
The **Close** button will close the Command window.

Help
The **Help** button will invoke Command window help.

Scroll Box
The **Scroll Box** button will allow you to scroll through any previously entered commands.

Command Line Overrides

An override is an optional command parameter that allows the user to temporarily override the established global (default) parameter. The overrides available for the Command window vary from system to system. For a detailed description of the Command window overrides available for your particular system, see Chapter 4 of this manual.

Using Symbols In The Command Window

Symbols can be entered in lieu of addresses or data values in Command window commands by typing a colon (:) followed by the symbol name. The value associated with the symbol name will be substituted in the command. Also, a number can be added to or subtracted from a symbol value by following the symbol with "+value" or "-value", respectively, where "value" is the number to be added.

The ':' command is used for symbol table display and editing. The 'symbol', '=', and 'value' portions of this command are all optional. This command comes in many different formats. Each of these is discussed in the sections that follow:

:
':' given by itself on a line causes SourceGate to display the entire contents of its internal symbol table:

:symbol
':' followed by a symbol will cause SourceGate to search for and display the first occurrence of the named symbol in the internal symbol table.

:symbol*
':' followed by a symbol with a wildcard character all occurrences of the symbol name including wildcards. For example, if ":test*" is entered, SourceGate will display all symbols of this type such as "test", "test1", "testit", etc.

:module/symbol
':' followed by a module name, followed by a '/', followed by a symbol name will cause SourceGate to search for the named symbol in or following the named module in the symbol table. If the symbol is found, it will be displayed.

:symbol=value
:module/symbol=value
These ':' commands perform symbol lookup as outlined above, but when the named symbol is found, the old value of the symbol is discarded, and the given value substituted for it.

In the above commands, 'value' defaults to a hexadecimal number unless one of the following characters is used to prefix the constant's value:

%	- Binary override prefix
@	- Octal override prefix
&	- Decimal override prefix
.	- Decimal override prefix
$	- Hexadecimal override prefix
0x	- Hexadecimal override prefix

If the command ":SYM100=0" was entered, the following examples show what the output would be when one of the optional prefixes is used:

:SYM100	SYM100 plus zero offset (output = 0)
:SYM100+10	SYM100 plus 10 hex (output = 10H)
:SYM100+.16	SYM100 plus 16 decimal (output = 10H)
:SYM100+%1000	SYM100 plus 1000 binary (output = 8H)
:SYM100+@10	SYM100 plus 10 octal (output = 8H)

In addition, the symbol can be defined to be the name of another symbol, preceded by a colon. If the named symbol does not exist it will be created. If 'module' is given, the symbol will be created in the named module, otherwise a dummy module name of "USER_DEFINED_MODULE" is assumed.

Using Module Commands In The Command Window

Module commands can be performed while in the Command window. These commands influence the information that is shown in the Command window. The following commands are used to enable a particular level of debugging on a module or globally:

:=A [module] (Assembly Mode)
The ':=A' command sets the named module (or all modules if none is named) to Assembly Level Debug mode. This is the default mode for all modules when SourceGate is started. While in this mode, only assembly language will be shown in the output of any Command window command, and all source code information will be suppressed.

:=C [module] (Clear Symbol Table)
The ':=C' command clears all symbols from the internal symbol table for the named module (or all modules if none is named). The symbol information for the cleared module must be read back into the internal symbol file in order to perform symbolic/source debugging on that module.

:=D [module] (Display Current Modes)
The ':=D' command displays source debug information for the named module (or all modules if none is named). Source debug information consists of the line number map acquired by SourceGate through examination of the module's line number symbols (whose format is set with the SourceGate command line switch '-L' or through the Source Code Interface Options window), the module's source code file name, and the current debug mode set for that module (Assembly, Mixed, or Source-only).

:=M [module] (Mixed Mode)
The ':=M' command sets the named module (or all modules if none is named) to Mixed Debug mode. Once a module has been enabled for mixed debug mode, that module's source file will be opened, and the source lines for that module will be inserted at appropriate points into the output of all commands which disassemble code (i.e. the 'L' (List) command, 'SS' (single-step) command, etc.).

When the ':=M' command is used, SourceGate must be able to locate and successfully open each named module's source code file. These source code files are assumed to reside in the current directory, and their names are assumed to be the module's name, with the extent of ".C".

If any of these assumptions are false, SourceGate will prompt for a source code file name, giving the name of the module for which it cannot find the source code file. The full path name of the file, along with the corresponding drive designator, should be entered at this time.

If no source code file is available, entering no file name at the prompt will cause SourceGate to set that module back to Assembly mode debug (which requires no source code file), and continue on to the next module to be set to Mixed debug mode.

:=S [module] (Source Mode)
The ':=S' command sets the named module (or all modules if none is named) to Source-only Debug mode. When source-only debug mode is selected for a particular module, all assembly language displays for that module will be suppressed. (Some assembly may be displayed in the source only mode. This assembly is from the library routines of the compiler and is source code to the emulator since most of these routines are written in assembly language). In addition, that module's source file will be opened and the source lines for that module will be inserted into the output of all commands which disassemble code (i.e. the 'L' (list) command, 'SS' (Single-Step) command, etc.).

When the ':=S' command is used, SourceGate must be able to locate and successfully open each named module's source code file. These source code files are assumed to reside in the current directory, and their names are assumed to be the module's name, with the extent of ".C".

If any of these assumptions are false, SourceGate will prompt for a source code file name, giving the name of the module for which it cannot find the source code file. The full path name of the file, along with the corresponding drive designator, should be entered at this time.

If no source code file is available, entering no file name at the prompt will cause SourceGate to set that module back to Assembly mode debug (which requires no source code file), and continue on to the next module to be set to Source debug mode.

Setting Up Module Search Paths and Source File Names

In past versions of SourceGate, users had some problems when they tried to use source level debugging on source files that were not located in the current directory. They either had to put the path of all of their source files in the PATH environment variable, our they had to enter the path for each file when source-level debugging was enabled. This was cumbersome to say the least. SourceGate II takes care of this problem by providing commands that allow you to define these paths when you invoke source-level debugging. You can implement these commands in a command file and run this command file when you wish to perform source-level debugging. An overview of these new commands is given in the sections that follow.

You can define the names of source files, along with search paths for these files, by implementing the following command:

:=m [modules] [-f <file>] [-p <path>]

This command can also be used for :=a and :=s. The parameters shown in this command are defined as follows:

[modules]	the modules that the command is to be performed on
-f <file>	sets the source file name for every module in [modules] to be <file>
-p <path>	searches for the module's source file in the specified <path>. A ':' is used to separate directories in a UNIX system, ';' in a Windows based system. This is the same format that is used in the path environment variable.

Some examples follow:

:=m -f /foo/bar.c

Sets EVERY module's source file to the SAME /foo/bar.c file.

:=m mod1 -f foo.c

Sets mod1's source file to foo.c, even if mod1 already had a different source file.

:=m mod1 -f foo.c -p /tmp:/usr:/bar (UNIX)

Sets mod1's source file to foo.c but it will also look for foo.c in each of the directories listed in the <path>.

If you knew ALL of the directories that your source files were in, you could do the following:

:=m -p <all of the directories separated by :>

This would search all of the directories listed for the source files if they could not be located in the current directory.

There is also a new Environment variable and a new section in the <processor>.INI file that SourceGate II uses to determine where to look for source files. When searching for source files, SourceGate II first looks in the current directory. If the file is not there, it then looks in the <processor>.INI file for a "HMISPATH" section. If it can't find this section, it then looks for the HMISPATH environment variable. If it can't find this environment variable, it then looks in the directories defined in the PATH.

In the HMISPATH environment variable and in the HMISPATH section in the .INI file, an asterisk can be placed at the end of a directory. This directs SourceGate II to search the subdirectories of that directory. For example, "C:\HMI*" will search this directory and all of its subdirectories.

The HMISPATH section in the system specific .INI file has subsections called "path#" where "#" should be replaced with 0, 1, 2, etc.

For example:

```
[HMISPATH]
path0=V:\RAND\HOME\*
path1=C:\
path2=C:\SOURCE\*
```

An added advantage to using the above .INI solution is that you don't have to use additional environment space. The HMISPATH size limitation has been changed to 1024 on DOS and UNIX (previously, it was 80 on DOS and 1024 on UNIX). Also, each "path#" may be up to 1024 characters in the .INI file.

If the system specific .INI file (e.g. 68000.INI) does not exist, it can be created with an ASCII text editor. Then, input the [HMISPATH] information into this file.

Creating Automated Command Files From The Command Window

To begin recording an automated command session file that can be played back later, choose "File/Record...". This activates the Command window and prompts you to enter the name of the file in which to store the recorded information. After you have entered the required filename, any commands that are typed in the Command window are recorded in the file until you choose "File/Stop Recording". The file is then closed and can be played back by choosing "File/Execute".

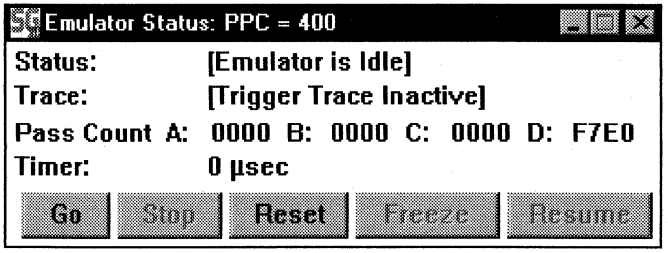
Note: This feature only is supported in the Command window and will not record operations in other windows.

Using The Emulator Status Window

The Emulator Status window allows you to view the emulation status using one window. The information shown in the border of this window includes the current program counter information or, if emulation is in progress, "RUNNING" is displayed. Inside of this window, the current emulation status (running or idle), trace trigger status (active or inactive), pass count information for all four events (200 Series emulators only), timer/counter fields (SPS-2000 Series emulators only), and interval timer reading are all displayed. In addition, there are buttons available that allow you to begin or stop emulation, reset the processor, and freeze or resume the trace buffers.

Opening The Emulator Status Window

To open the Emulator Status window, choose 'Run/Emulator Status'. An alternate method of opening the Emulator Status window is to use the accelerator key combination "Ctrl+E". The Emulator Status window has the following format:



An overview of each of the fields shown in this window follows.

Emulator Status Window Border

The Emulator Status window border shows the current Program Counter (PC) value when the system is idle. When the system is running, "RUNNING" appears in the border. If the Emulator Status window is minimized, the icon is shown with a label of "PC=XXXX" where "XXXX" is the current PC value.

Emulator Status Window 'Status' Field

The Emulator Status window 'Status' field will show the current status of the system. If the system is emulating, this field will display "Emulator is running". If the system is idle, this field will display "Emulator is idle".

Emulator Status Window 'Trace' Field

The Emulator Status window 'Trace' field will show the current status of the system's trace trigger. If the trace trigger condition plus delay has been reached, the message "Trace Trigger" will be displayed in this field. If the trace trigger condition plus delay has not been reached, the message "Trace Trigger Inactive" will be displayed.

Emulator Status Window 'Pass Count' Fields *(Not Available On SPS-2000 Series Emulators or BMDs)*

The Emulator Status window 'Pass Count' fields show the pass counts for events A, B, C, and D. The pass counts show how many times the defined event condition has occurred since the beginning of emulation and are updated in real time.

Emulator Status Window 'Time/Count' Fields *(SPS-2000 Series Emulators Only)*

The Emulator Status window 'Time/Count' fields show the current timer values for TC1, TC2, TC3, and TC4. If these Timer/Counters are enabled in the Do List Window, they will be updated in this window during emulation and will show the last recorded value when emulation is not in progress. If these Timer/Counters are not enabled in the Do List Window, they will show the current value for each Timer/Counter that is defined in the Timer/Counter window during emulation and will show the last recorded value when emulation is not in progress.

Emulator Status Window 'Timer' Field

The Emulator Status window 'Timer' field shows the time associated with the interval timer if it is active. If the interval timer is disabled, 0 microseconds will appear in this field. When the timer is active, this field will be updated in real time until the interval timer stop condition is reached. The format of the displayed value is mm:ss.cccuuu where 'mm' is minutes (71 is the maximum), 'ss' is seconds, 'ccc' is milliseconds, and 'uuu' is microseconds.

Emulator Status Window Buttons

The following buttons are available in the Emulator Status window:

Go

The **Go** button will begin emulation. This is the same command as the "G" command issued from the Command window. Note that this button is grayed out when emulation is in progress. **The "Go" button should not be used to start emulation from the CodeView windows. Use the CodeView "Start" button instead.**

Stop

The **Stop** button will stop emulation. This is the same command as the "SP" command issued from the Command window. Note that this button is grayed out when emulation is not in progress.

Reset
The **Reset** button will reset the CPU. This is the same command as the "RS" command issued from the Command window.

Freeze
The **Freeze** button will freeze the trace buffer. Note that this button is grayed out when the Freeze trace has been activated.

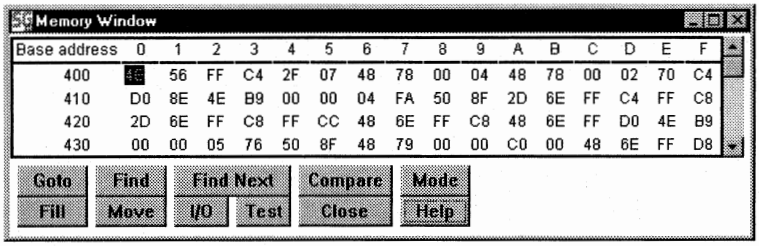
Resume
The **Resume** button will resume tracing. Note that this button is grayed out when the Freeze trace has not been activated.

Using The Memory Window

The memory window allows the user to display and change individual and block memory locations. This window is updated any time the emulator is single-stepped and when emulation is stopped.

Opening The Memory Window

To open the Memory window, choose 'Display/Memory'. An alternate method of opening the Memory window is to use the accelerator key combination "Ctrl+M". The Memory window has the following format:


Memory Window
Base address 0 1 2 3 4 5 6 7 8 9 A B C D E F
400 56 FF C4 2F 07 48 78 00 04 48 78 00 02 70 C4
410 D0 8E 4E B9 00 00 04 FA 50 8F 2D 6E FF C4 FF C8
420 2D 6E FF C8 FF CC 48 6E FF C8 48 6E FF D0 4E B9
430 00 00 05 76 50 8F 48 79 00 00 C0 00 48 6E FF D8
Goto
Find
Find Next
Compare
Mode
Fill
Move
I/O
Test
Close
Help

The "Tab" key advances to the next memory location and the "Shift-Tab" key combination retreats to the previous memory location. The arrow keys can be used in a similar manner. This allows a way to scroll the display to see those locations that do not reside in the current Memory window display.

Memory locations can also be viewed via the Command window using the "Dump" (D) command. For additional information, see ***Using The Command Window***. An overview of the Memory window follows.

Memory Window Buttons

The following buttons are available in the Memory window:

Goto

The **Goto** button allows the user to specify the base address of the memory display. For additional information, see ***Viewing Specific Memory Locations***.

Find

The **Find** button allows the user to search a block of memory for a particular data value. For additional information, see ***Searching Memory For A Specific Value***.

Find Next

The **Find Next** button allows the user to perform the search operation again, using the search value previously entered with the **Find** button. For additional information, see ***Searching Memory For A Specific Value***.

Compare

The **Compare** button allows the user to compare two blocks of memory to see how they differ in content. For additional information, see ***Comparing Memory Blocks***.

Mode

The **Mode** button allows the user to choose the display mode. For additional information, see ***Changing The Memory Window Display Mode***.

NOTE: This command is not available for all systems. If this button is grayed, your system does not support it.

Type (8051 Emulators Only)

The **Type** button allows the user to choose the memory type that is to be displayed. For additional information, see ***Changing The Memory Window Display Mode***.

Fill

The **Fill** button allows the user to fill a range of memory locations with a constant value. For additional information, see ***Filling A Memory Block With A Constant Value***.

Move

The **Move** button allows the user to move a block of memory to another location. For additional information, see ***Moving A Memory Block***.

I/O
The **I/O** button allows the user to read or write a particular memory location. This can be very useful when testing memory mapped I/O locations. For additional information, see ***Accessing Memory Mapped I/O Locations***.

NOTE: This command is not available for all systems. If this button is grayed, your system does not support it.

Test
The **Test** button allows the user to perform a nondestructive memory test on any RAM memory location. For additional information, see ***Testing A Memory Block***.

Close
The **Close** button closes the Memory window.

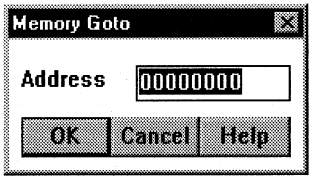
Help
The **Help** button invokes help for the Memory window.

Viewing Specific Memory Locations

To view specific memory locations in the Memory window, press the "Goto" button which will open the Memory Goto window. This window is described as follows:

Memory Goto Window

The Memory Goto window will allow the user to change the address for the beginning of the Memory window display:



After an address value is entered, the Memory window display will begin at the specified address.

Memory Goto Window Buttons

The following buttons are available in this window:

OK

The **OK** button will make the base address shown in the Memory window to be the address entered in the Memory Goto window.

Cancel

The **Cancel** button will cancel the goto operation and close the Memory Goto window.

Help

The **Help** button will provide help for this window.

Modifying Specific Memory Locations

A specific memory window location can be modified in one of several ways. Clicking once on the desired location will insert the edit bar cursor at that location and the selected location will change color to reflect the selection. Memory values can then be inserted by typing a new value. Current contents can be deleted by pressing the "Delete" key. Multiple values in a memory location can be selected by clicking once, holding the left-mouse button down, and then dragging over the selected values. Double-clicking selects the entire memory location. Once selected, these values can then be modified.

The "Tab" key advances to the next memory location and the "Shift-Tab" key combination retreats to the previous memory location. The arrow keys can be used in a similar manner.

Memory locations can also be modified via the Command window using the "Examine" (E) command. For additional information, see Chapter 4 of this manual.

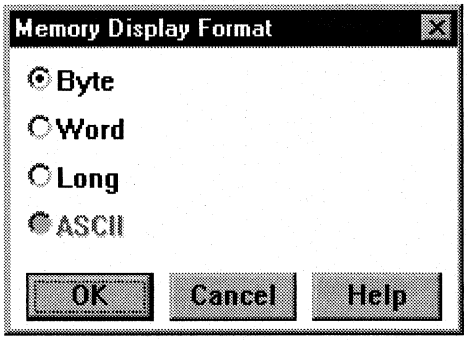
Changing The Memory Window Display Mode

To change the display mode of the Memory window display, press the "Mode" button. The Memory Display Format window will then be displayed.

Memory locations being viewed via the Command window can be displayed in different formats using the "Byte Mode" (BM), "Word Mode" (WM), and "Long Word Mode" (LM) commands. For additional information, see Chapter 4 of this manual.

Memory Display Format Window

The Memory Display Format window will change how the memory locations appear to the user:



Depending on the system being used, the choices for the display modes are "Byte", "Word", and "Long Word". The "Byte" mode will display each location as containing two hexadecimal characters, "Word" mode as containing four, and "Long Word" mode as containing eight.

Some systems may not display all of these modes. If any of these display modes are grayed, your system does not support them. The 200 Series 8051 emulator substitutes a "Format" button in place of the "Mode" button. This allows the different memory spaces of that device to be selected for viewing. Note that the "ASCII" mode is not available in this release of SourceGate and is grayed.

Memory Display Format Window Buttons

The following buttons are available in this window:

OK

The **OK** button will re-display the Memory window in the chosen display format.

Cancel

The **Cancel** button will cancel the format change operation and close the Memory Display Format window.

Help

The **Help** button will provide help for this window.

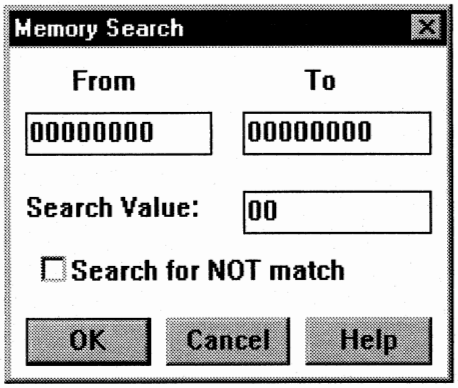
Searching Memory For A Specific Value

To search for a specific value in memory, press the "Find" or "Find Next" buttons. The Memory Search window will then be displayed.

Memory locations can also be searched via the Command window by using the "Search" (SR) and "Search Not" (SRN) commands. For additional information, see Chapter 4 of this manual.

Memory Search Window

The Memory Search window will allow the user to enter a value to be searched for in memory:



The Memory Search window is invoked when the "Find" button is pressed from within the Memory window. This window allows the user to search memory for a particular value. The user can also search for when a value is 'NOT' in memory. This can be a handy feature when trying to verify that memory was initialized correctly.

The 'From' field defines the address to begin the search from, the 'To' field defines the address to search to, and the 'Search Value' field defines the value to be searched for. The 'Search For NOT Match' box, when checked, will search memory and display those locations that do NOT contain the value contained in the 'Value' field.

When a match to the search is found, the matching location is highlighted in the Memory window. If no match is found, the message "Value not found" will be displayed.

Memory Search Window Buttons

The following buttons are available in this window:

OK

The **OK** button will begin the search operation.

Cancel

The **Cancel** button will cancel the search operation and close the Memory Search window.

Help

The **Help** button will provide help for this window.

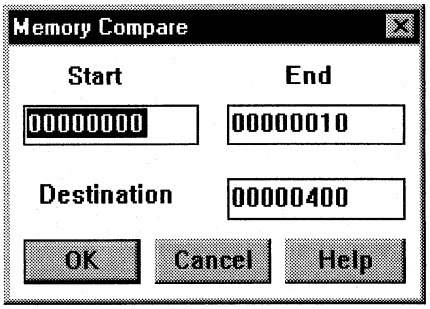
Comparing Memory Blocks

To compare two blocks of memory, press the "Compare" button. The Memory Compare window will then be displayed.

Memory locations can also be compared via the Command window by using the "Compare" (C) command. For additional information, see Chapter 4 of this manual.

Memory Compare Window

The Memory Compare window will allow the user to compare two blocks of memory and see any differences between them:



The Memory Compare window is invoked when the "Compare" button is pressed from within the Memory window. This window allows the user to compare two memory blocks to see any differences between them.

The 'Start' field defines the starting address of the first block, the 'End' field defines the ending address of the first block, and the 'Dest' field defines the beginning address of the block to be compared.

The results of this operation are shown in the Memory Output window. For additional information, see ***Memory Output Window***.

Memory Compare Window Buttons

The following buttons are available in this window:

OK

The **OK** button will begin the compare operation.

Cancel

The **Cancel** button will cancel the compare operation and close the Memory Compare window.

Help

The **Help** button will provide help for this window.

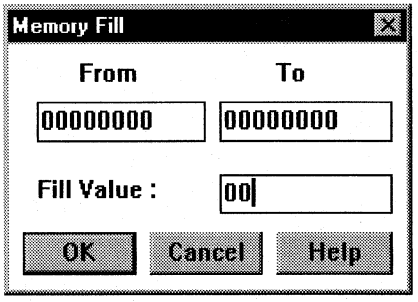
Filling A Memory Block With A Constant Value

To fill a memory block with a constant value, press the "Fill" button. The Memory Fill window will then be displayed.

Memory locations can also be filled via the Command window by using the "Fill" (F) command. For additional information, see Chapter 4 of this manual.

Memory Fill Window

The Memory Fill window will allow the user to fill a block of memory with a constant value:



The Memory Fill window is invoked when the "Fill" button is pressed from within the Memory window. This window allows the user to fill memory with a constant value.

The 'From' field defines the starting address, the 'To' field defines the ending address, and the 'Fill Value' field defines the value to be loaded into this range of memory addresses.

For example, if 'From' contained 5000H, 'To' contained 6000H, and 'Fill Value' contained 33H, every location from 5000H to 6000H would contain 33H upon execution of this command.

Memory Fill Window Buttons

The following buttons are available in this window:

OK

The **OK** button will begin the fill operation.

Cancel

The **Cancel** button will cancel the fill operation and close the Memory Fill window.

Help

The **Help** button will provide help for this window.

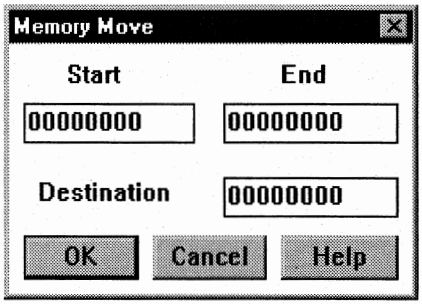
Moving A Memory Block

To move a memory block to another location in memory, press the "Move" button. The Memory Move window will then be displayed.

Memory blocks can also be moved via the Command window by using the "Move" (M) command. For additional information, see Chapter 4 of this manual.

Memory Move Window

The Memory Move window will allow the user to move a block of memory to another location in memory:



The Memory Move window is invoked when the "Move" button is pressed from within the Memory window. This window allows the user to move a memory block to another location in the memory map.

The 'Start' field defines the starting address of the block to be moved, the 'End' field defines the ending address of the block to be moved, and the 'Destination' field defines the address where the beginning of the block will be loaded.

For example, if 'Start' contained 5000H, 'End' contained 6000H, and 'Destination' contained 7000H, every location from 5000H to 6000H would be moved to its corresponding location beginning with address 7000H.

Memory Move Window Buttons

The following buttons are available in this window:

OK

The **OK** button will begin the move operation.

Cancel

The **Cancel** button will cancel the move operation and close the Memory Move window.

Help

The **Help** button will provide help for this window.

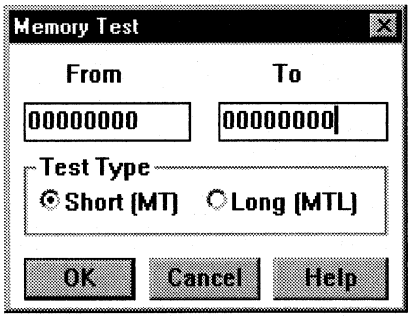
Testing A Memory Block

To test a block of memory, press the "Test" button. The Memory Test window will then be displayed.

Memory locations can also be tested via the Command window by using the "Memory Test Short" (MT) and "Memory Test Long" (MTL) commands. For additional information, see Chapter 4 of this manual.

Memory Test Window

The Memory Test window will allow the user to test a block of memory:



The Memory Test window is invoked when the "Test" button is pressed from within the Memory window. This window allows the user to perform a nondestructive memory test on any RAM location. **Note that ROM locations cannot be tested via this method.**

The 'From' field defines the first memory location to be tested, and the 'To' field defines the last memory location to be tested. The 'Test Type' field allows the user to define the type of test being performed. The 'Short(MT)' test writes an alternating pattern of AA (hex) and 55 (hex) and the 'Long(MTL)' test writes a 'Walking 1' pattern to each location. The test reads back the values that were written to each memory location in the defined range and checks to make sure that the memory is working properly.

The results of this operation are shown in the Memory Output window. For additional information, see ***Memory Output Window***.

Memory Test Window Buttons

The following buttons are available in this window:

OK

The **OK** button will begin the test operation.

Cancel

The **Cancel** button will cancel the test operation and close the Memory Test window.

Help

The **Help** button will provide help for this window.

Accessing Memory Mapped I/O Locations

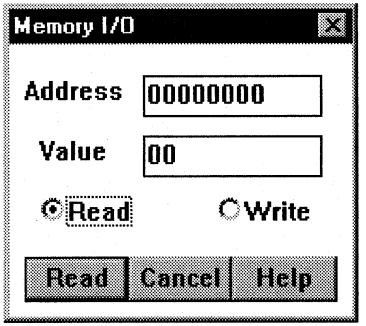
To test memory mapped I/O locations, press the "I/O" button. The Memory I/O window will then be displayed.

Memory mapped I/O locations can also be tested via the Command window by using the "Input" (I) and "Output" (O) commands. For additional information, see Chapter 4 of this manual.

Note: This feature is not supported on all systems. If this button is grayed, your system does not support this feature.

Memory I/O Window

The Memory I/O window will allow the user to either read or write memory mapped I/O locations:



The 'Address' field defines the I/O address where the read or write operation will be performed. The 'Value' field defines the value to be written to the defined address if 'Write' is selected. The 'Value' field will show the contents of the I/O location if 'Read' is selected.

Memory I/O Window Buttons

The following buttons are available in this window:

Read

The **Read** button will read the defined I/O location and display the results in the 'Value' field. This button is only shown when the "Read" radio button is selected.

Write

The **Write** button will write the defined I/O location with the value located in the 'Value' field. This button is only shown when the "Write" radio button is selected.

Cancel

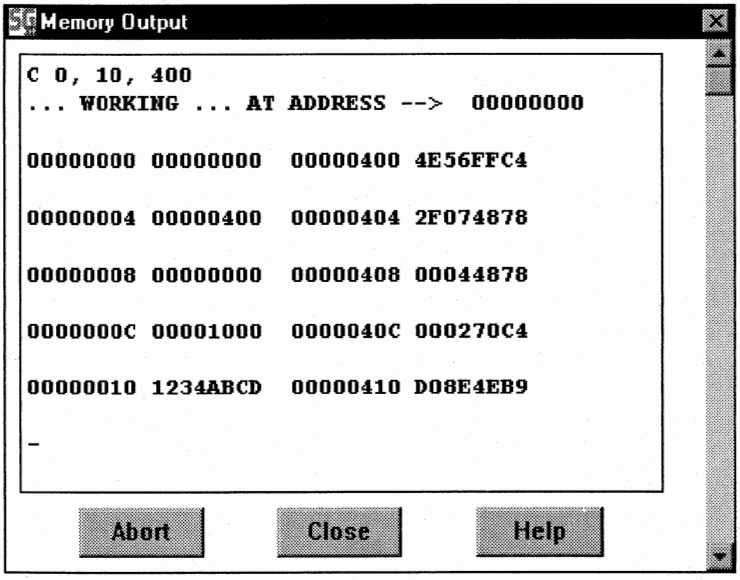
The **Cancel** button will cancel the I/O operation and close the Memory I/O window.

Help

The **Help** button will provide help for this window.

Memory Output Window

The Memory Output window shows the result of a Memory Compare or Memory Test operation:



If the previous operation was a "Compare" operation, the output is of the format:

Addr1 Val1 Addr2 Val2

where:

Addr1	is the first address in block 1 which differs
Val1	is the value of the block 1 location
Addr2	is the corresponding address in block 2 which differs
Val2	is the value of the block 2 location

If the previous operation was a "Test" operation, the output is of the format:

Addr Val1 Val2

where:

Addr	is the address where the memory test failed
Val1	is the value that was written to memory
Val2	is the bad value that was read from memory

Note that a "WORKING AT ADDRESS ----> XXXXXXXX" message may be displayed in certain instances.

Memory Output Window Buttons

The following buttons are available in this window:

Abort
The **Abort** button will abort the current operation.

Close
The **Close** button will close this window.

Help
The **Help** button will provide help for this window.

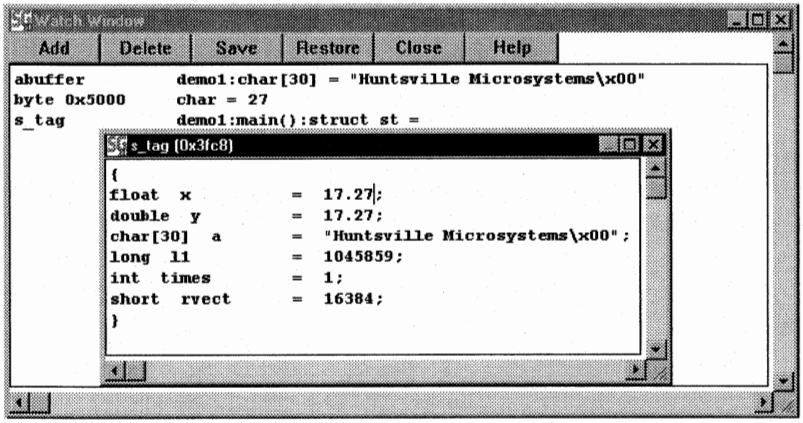
Using The Watch/Watch In Real Time Windows

The Watch and Watch In Real Time windows allow certain memory locations or variables to be "watched" and/or modified by the user. The Watch window allows the user to monitor and/or modify variables while single-stepping or any other time when emulation is halted. The Watch In Real Time window is used to monitor variables in real-time while emulation is in progress. The operation of both of these watch windows is identical.

Data structure information, along with the display of local variables, is supported when IAR Systems, Software Development Systems (SDS), or any other compiler that can produce IEEE-695, Common Object File Format (COFF), or ELF/DWARF output files is used. Watch windows can be saved to disk and loaded again at a later date. All variable types are supported and can be displayed.

Once a watch variable is being displayed in the Watch or Watch In Real Time window, you can click on the variable line to see more detailed information about the watch variable. For example, if a data structure is being shown, clicking on the data structure watch variable will open an additional window that only contains information about that data structure. If you continue to click on individual elements of the structure, additional windows will be shown for these individual elements and they will automatically be added as watch variables in the Watch window. Arrays are handled in a similar manner so that individual elements of the array can be displayed if so desired.

An example Watch window is shown below:



Opening The Watch/Watch In Real Time Windows

To open the Watch window, choose "Display/Watch". An alternate method of opening the Watch window is to use the accelerator key combination "Ctrl+W". To open the Watch in Real Time window, choose "Display/Watch In Real Time".

Note: In order to access the Watch In Real Time window, the system must be equipped with Shadow RAM and the Shadow RAM must be configured correctly. For additional information on configuring Shadow RAM, see Chapter 4 of this manual. Not all systems support the real-time watch feature. If the "Display/Watch In Real Time" menu item is grayed out, this feature is not supported for your system.

An overview of the Watch/Watch In Real Time windows follows.

Watch/Watch In Real Time Window Buttons

The following buttons are available in the Watch/Watch In Real Time windows:

Add
Add a variable to the Watch window. For additional information, see ***Adding A Watch Variable***.

Delete
Remove the highlighted variable from the Watch window. For additional information, see ***Deleting A Watch Variable***.

Save
Save the current Watch window to a file. For additional information, see ***Saving A Watch Variable***.

Restore
Restore a previously saved Watch window. For additional information, see ***Restoring A Saved Watch Variable***.

Close
Close the current Watch window.

Help
Invoke Watch window help.

Adding a Watch Variable

Variables can be added to the Watch window by clicking on the "Add" button or by entering in the variable information manually. The following topics provide information on how to add watch variables using each of these methods.

Entering Watch Variables Manually In The Watch Window

The Watch window allows the user to enter watch variables manually by clicking on a line in the watch window and typing in the desired type, address, and display type information for the variable to be "watched" in the format of:

type override address display_type

Pressing the "Enter" key will direct SourceGate II to accept the watch variable as a new entry and its value will then be updated. If the variable to be watched is a pre-defined symbol, the user can bypass the above step by typing the name of the symbol in the watch entry field and pressing the "Enter" key. SourceGate will search the symbol table information and display the variable in its proper form.

Probably the easiest way to enter a watch variable that is a symbol name is to press the right mouse button while in the watch window. This will invoke the Symbol Selection window from which a symbol can be chosen by clicking on the desired variable and then pressing the "Select" button. Once the symbol has been selected, it will automatically be added to the watch variable list. See also Symbol Selection Window.

Each of the manual entry fields (type, override, address, and display_type) are described in the sections that follow.

Watching a Symbol

To choose a symbol name as the watch variable, make the Watch window the active window and then press the right mouse button. This will bring up the Symbol Selection window that shows a listing of all available symbols. Click on the desired symbol to watch and then press the "Select" button. The information for the chosen symbol will be automatically loaded in the Watch window. For additional information, see ***Working With Symbols***.

If the user wishes to watch certain elements of a structure, this can be accomplished by entering "structure.element". Similarly, if the user wishes to watch certain elements of an array, this can be accomplished by entering "array_name[element]". For multidimensional arrays, type "array_name[a][b]" to watch a particular element. Both structures and arrays will be displayed in separate "Watch Output" windows if the output if too large to be shown in the Watch window.

Watch Variable Types

The type field is used to tell the Watch window what type of variable is being specified. Any type can include an '*' after the type to denote a pointer.

Some examples follow:

char	Variable at the address is of type char
char*	Value is a pointer to a variable of type char
char**	Value is a pointer to a pointer that points to a variable of type char

The following types are available and should be entered as shown:

%<reg> (where <reg> is a processor register (case sensitive))
char
short
int
long
unsigned char
unsigned short
unsigned int
unsigned long
float
double
address
byte
word
char*
short*
int*
long*
float*
double*

Watch Variable Overrides

On some systems, the user can specify an override designator as part of an address. Overrides may precede the address in the form {override}address. For example, if the user wanted to watch an integer in the supervisor/data space of a 68000 based system at location 5000 hex and have it displayed as a decimal value, they should enter "integer {SD}0x5000 %d".

Watch Variable Addresses

The address field is used to tell the Watch window the address of the data to be watched. This field can be entered in decimal (default) or in hexadecimal by preceding the address with "0x" or "0X".

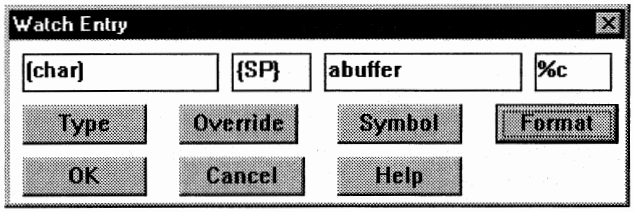
Watch Variable Display Types

The display type field is used by the Watch window to determine the way in which the data will be displayed. The default display type is decimal. All types must be preceded with "%", the case is ignored. An example of the available display types follows:

%c (Character)
%d (Decimal)
%u (Unsigned Decimal)
%f (Float)
%h (or x)(Hex)
%o (Octal)
%b (Binary)
%s (String)

Using The Watch Entry Window To Enter Watch Variables

To add a variable to the Watch window using the "Watch Entry" window, press the "Add" button. This will open the "Watch Entry" window:



While in this window, you can enter a watch variable by using the buttons shown in this window or by typing in the information manually. For information on entering Watch variables manually, see ***Entering Watch Variables Manually In The Watch Window*** in the previous section. The same parameters are used when entering variables manually in the Watch Entry window.

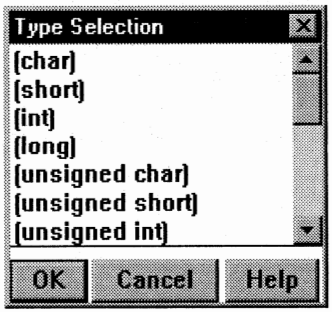
The following sections explain how to enter a variable using the buttons shown in this window.

Watch Entry Window Buttons

The Watch Entry window buttons can be used to enter the information for a watch variable definition. The "Type", "Override", "Symbol", and "Format" buttons invoke additional windows from which watch variable parameters can be defined. An explanation on how to use each of these buttons is defined in the sections that follow.

It should be noted that, if you are using buttons to enter watch variables, you should not enter any parameters manually until all of the desired buttons have been pressed. If you previously entered information manually and then press a button, the manually entered information will be deleted. To mix manual and button entries, always use the buttons first and then edit the watch entry line manually.

Type
The **Type** button will open the 'Type Selection' window as shown below:



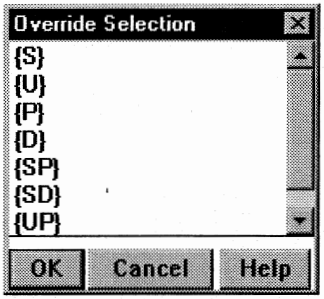
From this window, you can choose most of the available variable types. Additional types can be entered manually. To choose a type, click on it and then press the **OK** button. The selected type will then be shown in the 'Watch Entry' window. The pre-defined choices are as follows:

char
short
int
long
unsigned char
unsigned short
unsigned int
unsigned long
float
double
address
byte
word
char*
short*
int*
long*
float*
double*

If you decide not to enter a type, press the **Cancel** button. To obtain help for this window, press the **Help** button

Override

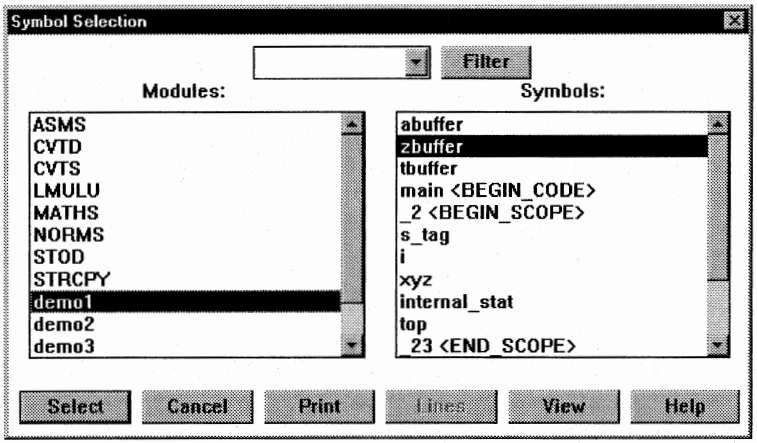
The **Override** button will open the 'Override Selection' window as shown below:



Overrides are used on certain systems to define a particular memory space in which the variable to be "watched" resides. The Override Selection window allows the user to choose a pre-defined override by using the mouse pointer and left mouse button to highlight the desired override. Pressing the **OK** button will load the defined override into the Watch Entry window. The overrides shown depend on the system being used. See Chapter 4 of this manual or the on-line help for additional information on the overrides available for your system.

Symbol

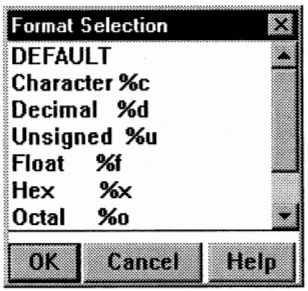
The **Symbol** button will open the "Symbol Selection" window as shown below:



A pre-defined symbol can be selected from this window and then "watched" in the Watch window. For additional information, see ***Working With Symbols***.

Format

The **Format** button will open the "Format Selection" window as shown below:



The Format Selection window allows the user to choose a pre-defined variable display format by using the mouse pointer and left mouse button to highlight the desired format. Pressing the **OK** button will load the defined format into the Watch Entry window. The **Cancel** button will cancel the format entry operation.

The output value for this watch variable will be in the specified format when displayed in the Watch window. Additional formats can be entered using the manual entry method. These choices represent the more common variable display formats. The pre-defined choices are as follows:

DEFAULT
%c (Character)
%d (Decimal)
%u (Unsigned Decimal)
%f (Float)
%h (or x)(Hex)
%o (Octal)
%b (Binary)
%s (String)

Note that DEFAULT will try and pick out the best display type based on the defined type (e.g. pointers will default to hex, characters to ASCII, integers to decimal, etc.).

OK
The **OK** button will enter the defined watch variable into the watch window.

Cancel
The **Cancel** button will cancel the watch variable entry procedure.

Help
The **Help** button will invoke help for this window.

Deleting a Watch Variable

Previously entered watch variables can be deleted. To choose the variable to delete, press the left mouse button on the variable to be deleted. This will place the cursor in the line containing the variable. Then, press the left mouse button while pointing to the "Delete" button. The selected watch variable will then be deleted.

Modifying Watch Window Variable Values

The user can modify the values of certain variables in a Watch window. All variable types can be modified with the exception of bit field and floating point variables. To modify a variable's value, simply highlight the value and then type in the new value. The variable's value will immediately be changed in the corresponding memory location.

Saving a Watch Window

It is possible to save a Watch window for future reference. To save a Watch window, press the left mouse button while pointing to the "Save" button. The user will then be prompted to enter a filename. The Watch window will then be saved as that file.

Restoring a Saved Watch Window

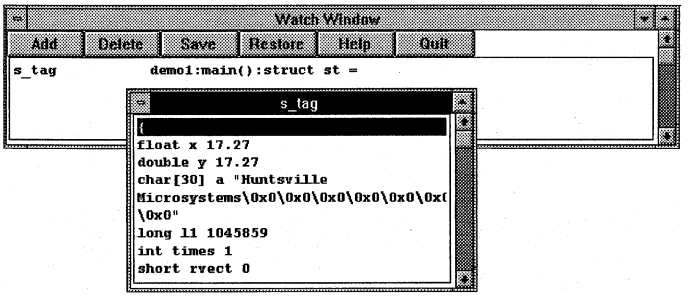
To restore a previously saved Watch window, press the left mouse button while pointing to the "Restore" button. The user will then be prompted to enter the filename of the Watch window to be restored. After choosing the name of the file, the named Watch window will be retrieved and the current values for its variables will be displayed.

Watch Window Examples

The following examples were included in order to help give you a better understanding of how to use the Watch window.

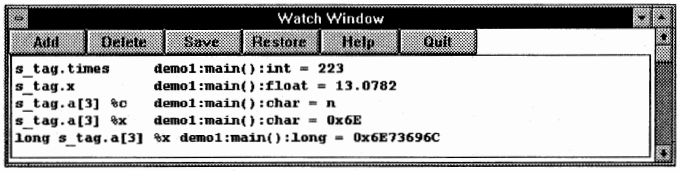
Watching A Structure

To watch an entire structure, enter the name of the structure as a watch variable. A separate window showing the contents of each element of the structure will be shown if the structure's information is too long to be shown in the Watch window:



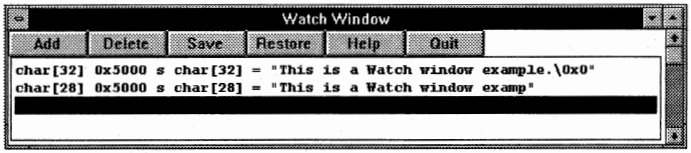
Watching An Element Of A Structure

If you wish to watch certain elements of a structure, this can be accomplished by entering "structure.element":



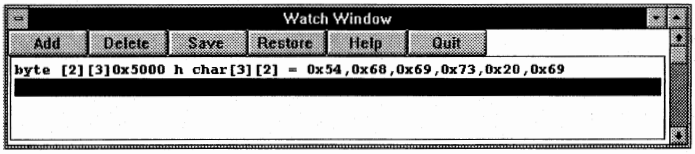
Watching A One-Dimensional Array

To watch a one-dimensional array, enter the array size. If you wish to watch certain elements of an array, this can be accomplished by entering "array_name[element]". The following is an example of a character array:



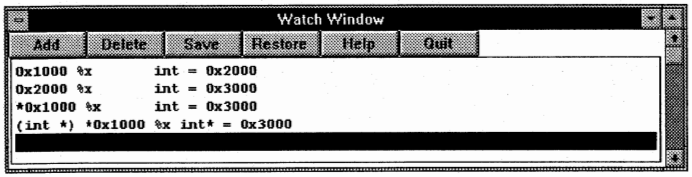
Watching A Multi-Dimensional Array

For multi-dimensional arrays, type "array_name[a][b]" to watch particular elements. Arrays will be displayed in a separate "Watch Output" window if the output if too large to be shown in the Watch window.



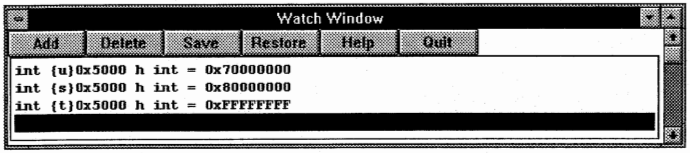
Watching Pointers

Pointers can be watched by preceding an address with the pointer designator "*".



Using Overrides In The Watch Window

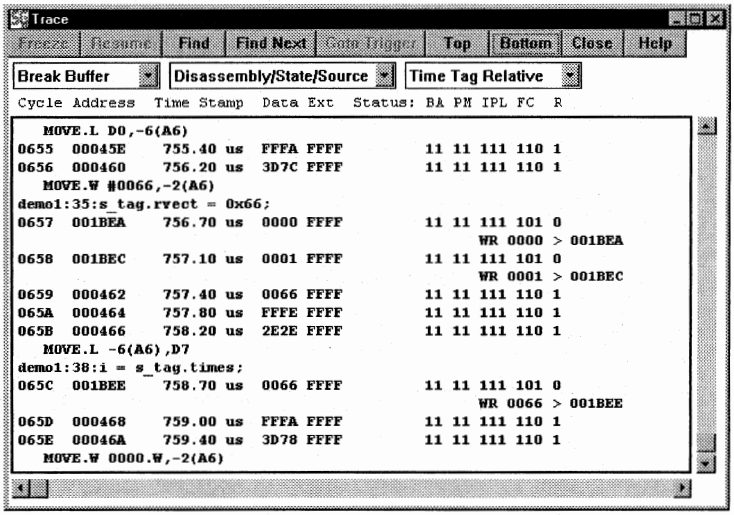
Overrides can be used to watch different memory spaces. In the example shown below, address 0x5000 in a 68030 system is being watched. The 68030 has separate Supervisor and User memory spaces. The {s} override is a 68030 override which specifies Supervisor memory space and the {u} override is a 68030 override which specifies User memory space. The {t} override specifies target memory. Notice that, using overrides, the contents of address 0x5000 is shown to be different for each memory space, which is to be expected since there are actually three distinct 0x5000 addresses. Without the use of overrides, only one space could be shown at a time.



Using The Trace Window *(N/A For BMDs)*

The Trace window allows the user to view and search through the two trace buffers (Trigger and Break) and to initiate and view the Freeze trace. The trace buffers can be displayed in a combination of disassembly, state, source, and time tag information (time tag information requires the Performance Analysis Card (PAC)). Certain emulators support an enhanced trace buffer that shows source code in a different color for ease of identification. Other functions of the Trace window include trace search and trace print.

The Trace window is updated each time emulation is terminated through a breakpoint, stop command, reset command, or when a freeze trace command is issued. An example Trace window is shown below:



The trace buffer for your system may look slightly different from the one shown above. Consult Chapter 4 of this manual for information specific to your particular system.

Opening The Trace Window

To open the Trace Window, choose "Display/Trace". An alternate method of opening the Trace window is to use the accelerator key combination "Ctrl+T".

When the trace window is open, an additional menu item labeled "Trace" is shown in the SourceGate II menu bar. This menu item will show the user an additional set of selections which allow the user to define what information is shown in the trace buffers.

The following topics provide additional information about the Trace window:

Capturing Trace Data

Trace data is captured any time emulation is started and then stopped via any method. This includes starting and stopping from the CodeView, Command, Go From/To..., and Emulator Status windows or via thebreakpoint features of the system.

The exception to the stop in emulation requirement is accomplished through the use of the "Freeze" trace command which allows a "snapshot" of the current trace buffer to be shown to the user without stopping emulation. The "Resume" command re-starts the trace record operation.

Choosing Between Break and Trigger Buffers

There are two trace buffers available to the user. Each of these buffers are FIFO (First In - First Out) types and each are either 4K, 16K, or 128K cycles deep depending on the system being used.

Since the system has both a 'Break' and a 'Trigger' buffer, you can specify which of these buffers to view in the Trace window. This is done by choosing 'Trace/Break Buffer' to view the Break buffer or 'Trace/Trigger Buffer' to view the Trigger buffer. On certain emulators, a scroll box is available on the left side of the trace display that allows you to select between these different buffers.

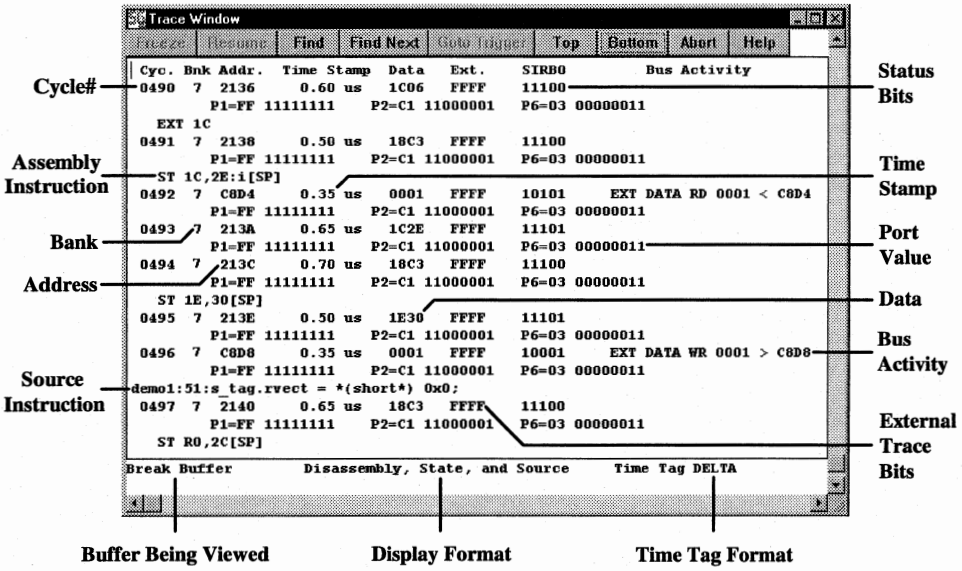
Note that the Break buffer will always be available but, for the Trigger buffer to exist, a valid trace trigger condition along with the specified trace delay as defined in the 'Trace Trigger' window must have occurred. (On SPS-2000 Series emulators, a Trigger buffer must be defined to exist, else the entire trace buffer will be configured as a Break buffer. See ***Using The Sequencer Windows*** in Chapter 4 of this manual for additional information on how to do this.) The Trigger buffer is the traced information which was recorded around the trigger point. This buffer will include any information recorded before the trigger point along with the number of cycles specified to be recorded after the trigger point.

It should be noted that no trigger buffer will exist if the conditions of the trigger point plus the delay value have not been met. If this occurs, the message "No data recorded" will be displayed in the trigger buffer display.

Also note that the buffer selection shown in the Trace menu depends upon the current buffer being viewed. If the Break Buffer is being viewed, the menu will show "Trigger Buffer" as a command. If the Trigger Buffer is being viewed, the menu will show "Break Buffer" as a command.

Description of Trace Display Fields

The trace display has several fields that are listed in column form. The fields that are shown depend on the format chosen for the trace display. An example Trace window is shown below with the available fields listed:


Trace Window
Freeze
Resume
Find
Find Next
Goto Trigger
Top
Bottom
Abort
Help
Cyc. Bnk Addr. Time Stamp Data Ext. SIRBO Bus Activity
0490 7 2136 0.60 us 1C06 FFFF 11100
P1=FF 11111111 P2=C1 11000001 P6=03 00000011
EXT 1C
0491 7 2138 0.50 us 18C3 FFFF 11100
P1=FF 11111111 P2=C1 11000001 P6=03 00000011
ST 1C,2E:i[SP]
0492 7 C8D4 0.35 us 0001 FFFF 10101 EXT DATA RD 0001 < C8D4
P1=FF 11111111 P2=C1 11000001 P6=03 00000011
0493 7 213A 0.65 us 1C2E FFFF 11101
P1=FF 11111111 P2=C1 11000001 P6=03 00000011
0494 7 213C 0.70 us 18C3 FFFF 11100
P1=FF 11111111 P2=C1 11000001 P6=03 00000011
ST 1E,30[SP]
0495 7 213E 0.50 us 1E30 FFFF 11101
P1=FF 11111111 P2=C1 11000001 P6=03 00000011
0496 7 C8D8 0.35 us 0001 FFFF 10001 EXT DATA WR 0001 > C8D8
P1=FF 11111111 P2=C1 11000001 P6=03 00000011
demo1:51:s_tag.rvect = *(short*) 0x0;
0497 7 2140 0.65 us 18C3 FFFF 11100
P1=FF 11111111 P2=C1 11000001 P6=03 00000011
ST R0,2C[SP]
Break Buffer
Disassembly, State, and Source
Time Tag DELTA
Cycle#
Assembly Instruction
Bank
Address
Source Instruction
Status Bits
Time Stamp
Port Value
Data
Bus Activity
External Trace Bits
Buffer Being Viewed
Display Format
Time Tag Format

Note that the trace buffer for your particular system will be similar, although some of the above fields may not be available.

An explanation of each of these fields can be found in the following topics:

Trace Cycle Field

The Cycle field shows the number associated with each memory cycle in the trace buffer. The first cycle in the trace buffer always begins with 0 unless the trace is being qualified. These numbers are sequential. However, if the user has chosen one of the non-state modes, there may be areas where cycle numbers numbers appear to be missing. This is due to only the assembly (or source) instruction being shown minus the state information. If one of the state modes is then chosen, these "missing" cycles will reappear in the trace display.

The cycle field will be shown for all trace display modes. The maximum number of this field is between FFF (hex) and 1FFFF (hex) depending on the system being used.

Trace Bank Field (N/A For Simulators)

Certain 8-bit emulators allow banking operations. When using these systems with banking enabled, a "Bank" bit is shown in the trace buffer. This allows the user to track what bank is being accessed on each processor cycle.

The bank field will be shown for all trace display modes in those systems that support banking operations. This field is not shown for systems that do not support banking operations. The value shown in this field can range from 0 - F depending on the system being used.

Trace Address Field

The Address field shows the address that was on the address bus for each cycle.

The address field will be shown for all trace display modes. The size of the address field is between 4 and 8 hexadecimal digits depending on the system being used.

Trace Data Field

The Data field shows the data that was on the data bus for each cycle.

The data field will be shown for any of the state display modes. The size of the data field is between 2 and 8 hexadecimal digits depending on the system being used.

Trace External Field (N/A For Simulators)

The External Trace Bits field shows the values of the external trace bits for each cycle. These bits plug into the "External Trace" connector of the emulator and are used to help qualify an event condition with an external condition (e.g. - a chip select going low in the target system).

The external trace bits field will be shown for any of the state display modes.

Trace Status Bit Field

The Trace Status Bits show processor specific signals that depend on the system being used. For additional information on this field, see Chapter 4 of this manual or reference the on-line help.

The trace status bits fields will be shown for any of the state display modes.

Trace Bus Activity Field

The Bus Activity field is shown in the far right column of the trace display and shows any reads or writes to memory that occur. The format of the activity field is:

Reads: RD {data read} < {address read from}

Writes: WR {data written} > {address written to}

The bus activity field will be shown for all trace display modes though the "Bus Activity" header will be suppressed for any of the state display modes.

Trace Time Stamp Field

If the emulator has a Performance Analysis Card (PAC) installed (Simulators provide a "simulated" PAC), and one of the "Timing" display modes is turned on, this field will show a time stamp value for each cycle.

The time stamp field will be shown for all trace display modes if one of the "Timing" modes is enabled. To enable one of the "Timing" modes, choose "Trace/Timing".

Trace 'Ports' Switch (N/A For Simulators)

Certain emulators allow port values to be displayed in the trace buffers. The value of the ports can be displayed for each cycle by selecting "View/Ports" from the Trace Menu Bar. When the "Ports" option is checked, port values are displayed for each cycle shown in the trace. These port values are displayed in both hexadecimal and binary.

If this switch option is grayed out, your system does not support the displaying of Ports.

Trace 'Filtered' Switch (N/A For Simulators)

Certain emulators allow the trace buffers to be "filtered". The trace buffer can be displayed as being "filtered" by selecting "View/Filtered" from the Trace Menu Bar. When the "Filtered" option is checked, unused internal cycles of the processor are filtered out of the trace buffer display.

If this switch option is grayed out, your system does not support the trace "filter" feature.

Formatting the Trace Display

The display format of the trace display can be chosen by using one of the "Trace/View" options. Certain systems contain an enhanced trace that provides list boxes in the trace window that allow you to choose the buffer to view (break or trigger), the trace display format, and time stamp format (if applicable). In addition, this enhanced trace will show source code lines in a different color for ease of identification.

An explanation of each of these options follows:

Trace Disassembly Format

The trace buffers will show only disassembly information which includes cycle, bank (if applicable), address, disassembled instructions, bus activity, and time stamp (if applicable). All state and source code information will be suppressed.

Trace State Format

The trace buffers will show only state information which includes cycle, bank (if applicable), address, data, external trace bits, status bits, bus activity, and time stamp (if applicable). All disassembly and source code information will be suppressed.

Trace Source Format

The trace buffers will show only high-level source information which includes the module name, line number of the module, the actual high-level line of code, bank (if applicable), and time stamp (if applicable). It should be noted that some disassembled instructions and bus activity may be shown also. This information usually appears when compiler routines are traced that were written in assembly language and, thus, have no high- level source code available for them. SourceGate has no choice but to display this information as the "source" associated with the displayed cycles.

Note that on systems equipped with the enhanced trace feature, the source code lines will be shown in a different color for ease of identification.

Trace Disassembly and State Format

The trace buffers will show both disassembly and state information which includes cycle, bank (if applicable), address, data, external trace bits, status bits, bus activity, and time stamp (if applicable). All high-level source information will be suppressed.

Trace Disassembly and Source Format

The trace buffers will show both disassembly and source information which includes cycle, bank (if applicable), address, module name, line number of the module, the actual high-level line of code, bus activity, and time stamp (if applicable). All state specific information will be suppressed.

Note that on systems equipped with the enhanced trace feature, the source code lines will be shown in a different color for ease of identification.

Trace Disassembly, State, and Source Format

The trace buffers will show all available information while in this mode which includes cycle, bank (if applicable), address, data, external trace bits, status bits, bus activity, module name, line number of the module, the actual high-level line of code, and time stamp (if applicable). No information is suppressed in this mode.

Note that on systems equipped with the enhanced trace feature, the source code lines will be shown in a different color for ease of identification.

Trace Timing Off Format

If the Performance Analysis Card (PAC) is installed in the emulator (Simulators provide a "simulated" PAC), a time stamp can be included for each cycle in the trace display. This option turns the time stamp off.

Trace Timing Relative Format

If the Performance Analysis Card (PAC) is installed in the emulator (Simulators provide a "simulated" PAC), a time stamp can be included for each cycle in the trace display. This option makes the time stamp to be relative to the beginning of the trace buffer.

Trace Timing Delta Format

If the Performance Analysis Card (PAC) is installed in the emulator (Simulators provide a "simulated" PAC), a time stamp can be included for each cycle in the trace display. This option makes the time stamp to be a delta value between each cycle.

Using the Freeze Trace Feature

The Freeze function allows the user to view the trace buffer and to modify the event and sequence definitions while the processor continues to run. While "freeze" is active, functions such as tracing, breaking, and triggering are disabled. However, if a Trigger buffer was captured prior to the "freeze" command being executed, the Trigger buffer will be shown when the "freeze" mode is entered. If a Trigger buffer was not defined or not captured prior to the "freeze" command being executed, a "snapshot" of the trace buffer will be shown reflecting the contents of the trace buffer when the "freeze" command was executed.

To activate the freeze mode, choose "Trace/Freeze" or press the "Freeze" button. To resume normal operation, choose "Trace/Resume" or press the "Resume" button.

Trace 'Goto' Commands

The following 'Goto' commands provide an easy way to move around in the trace buffers.

Trace/Goto Top

The 'Trace/Goto Top' command will load the trace display with the information contained at the beginning of the buffer. Note that the "Top" button can also be used to perform this function.

Trace/Goto Bottom

The 'Trace/Goto Bottom' command will load the trace display with the information contained at the end of the buffer. Note that the "Bottom" button can also be used to perform this function.

Trace/Goto Trigger

The 'Trace/Goto Trigger' command will load the trace display with the information around the defined trigger point. The cycle containing the trigger point is shown labeled as "--- Trace Trigger ---". Note that the "Goto Trigger" button can also be used to perform this function.

Trace Search Window

To search the trace buffer for a particular condition, choose "Search/Find" or press the "Find" button. The user can then enter the pattern of information to be searched for. The search condition can be either a certain cycle or a combination of address, bank (if applicable), data, external trace bits, and status bits.

The Cycle, Bank (if applicable), Address, Data, Port (if applicable), and External values can be changed by selecting the desired field and entering a value for that field. Previously entered values can be obtained by using the scroll box button to reveal and then select the desired value. Symbols can be entered for the Address and Data fields by pressing the right mouse button while the cursor is in one of these fields.

The status bit values can be changed by placing the mouse pointer on the desired status bit and pressing the left mouse button. This will toggle through the choices for the status bits. The choices are 0, 1, and 'X' where 'X' is the don't care condition. For additional information on the status bits that can be searched for in your particular system, see Chapter 4 of this manual.

After the search function is started, the entire trace buffer will be searched and the first cycle where the search condition is met will be highlighted.

To search for the next occurrence of the defined pattern, choose "Search/Repeat Last Search" or press the "Find Next" button.

Trace Search Window Buttons

The following buttons are available in the Trace Search window:

0,1,X

The user can toggle the status bit value for each status bit by moving the mouse pointer to the desired status bit and pressing the left mouse button. This will toggle through all possible choices for the status bit (0, 1, and X (Don't Care)).

OK

The **OK** button is used to accept the changes made in the search pattern and to begin the search.

Cancel

The **Cancel** button will cancel the changes made in the search pattern and will close the Trace Search window.

Clear
The **Clear** button will clear all entries in this window by setting their values to "X" (Don't Care).

Help
The **Help** button will invoke help for this window.

Scroll Box
The **Scroll Box** button will allow the user to select a previously entered value for that field.

Saving Trace Information to a File

To save the trace buffer information to a file, choose "File/Print". The trace buffer can then be saved to a file by choosing the "Print Output to File" option and then entering the desired filename to hold the trace information. The user can also define the beginning and ending cycle number which determines how much of the trace information is output.

Print Trace Buffer Window

The trace information can be printed to a printer or to a file. The "Start Cycle Number" field is the first cycle that will be printed and the "End Cycle Number" field is the last cycle that will be printed. If "Print Output to File" was selected, the trace information will be printed to the file defined in the "File Name" field. If "Print Output to Printer" was selected, the trace information will be printed to the currently defined printer.

Print Trace Buffer Window Buttons

The following buttons are available in this window:

OK
The **OK** button will begin the print operation.

Cancel
The **Cancel** button will cancel the print operation.

Help
The **Help** button will invoke help for this window.

Scroll Box
The **Scroll Box** buttons will allow the user to scroll through and select from previously entered values.

Trace Window Buttons

The buttons in the Trace window allow the user to perform certain actions. An explanation of all available Trace window buttons is given as follows:

Freeze
The **Freeze** button allows the user to view the trace and to modify events and sequences while the emulator continues to run.

Resume
The Resume button should be pressed when the trace is in freeze mode in order to return to the normal mode of emulation.

Find
The **Find** button should be pressed when the user wishes to search for a particular pattern in the trace buffer.

Find Next
The **Find Next** button will find the next occurrence of the defined search pattern.

Goto Trigger
The **Goto Trigger** button will find the trace trigger if it exists in the trace buffer.

Top
The **Top** button will take the display to the top of the trace buffer.

Bottom
The **Bottom** button will take the display to the bottom of the trace buffer.

Abort
The **Abort** button will stop the trace display from scrolling after the current scroll operation is completed. This button is handy if the user has entered multiple page up or page down operations and then decides to cancel the scrolling operation.

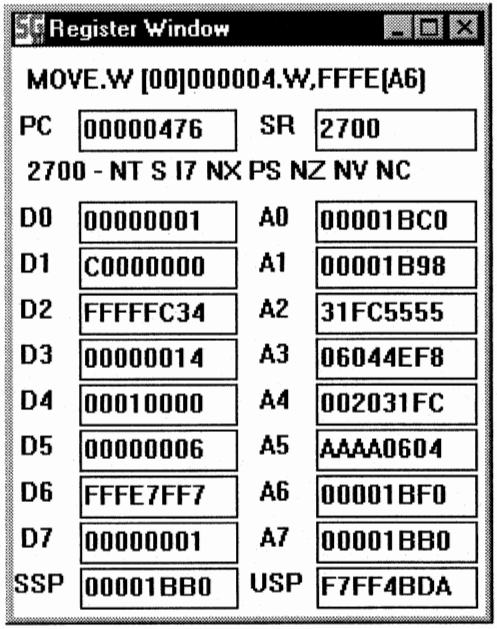
Help
The **Help** button will invoke help for this window.

Using The Register Window

The Register window shows the contents of the processor's registers including the status of the flags and the current disassembly mnemonic being pointed to by the program counter register. The register values are always accurate whenever emulation is not in progress, and are the last known value whenever emulation is in progress.

Registers can be altered in the register display window by selecting the desired register data field and then and entering a new value.

The Register window is automatically updated whenever the system is single-stepped, a breakpoint occurs, a stop command is executed, or the system is reset. The registers shown in the Register window are dependent on the system being used. An example Register window is shown below:



Opening The Register Window

To open the Register window, choose 'Display/Registers'. An alternate method of opening the Register window is to use the accelerator key combination Ctrl+R.

Modifying Register Values

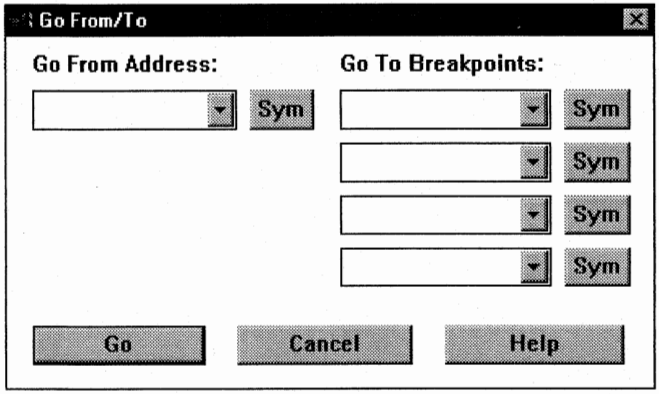
To modify a register, move the insertion bar to the register to be changed and then press the left mouse button. You can then enter the desired information beginning at the insertion point. To delete the old contents of a register and add a completely new value, click and drag to highlight the entire contents of the register and then enter the new value.

Available Registers

Consult the on-line help along with the databook of the processor manufacturer for a description of the registers specific to your system that are shown in the Register window.

Using The 'Go From/To...' Window

The 'Go From/To...' window allows the user to set temporary breakpoints without having to use the Command window. An example 'Go From/To...' window is shown below:



The fields shown in this window are described in the following sections.

'Go From/To...' Window 'Go From Address' Field

The 'Go From Address' field is the location where emulation will begin when the "Go" button is pressed. To enter a value, place the mouse cursor in this field and press the left mouse button. The user can then enter a hexadecimal beginning address in this field. A symbol can be entered by pressing the "Sym" button located next to the field and choosing the desired symbol. The "Scroll Box" button can be used to scroll through and select any previously entered symbols or addresses.

Once an address or symbol has been entered, pressing the "Go" button will begin emulation at that address.

'Go From/To...' Window 'Go To Breakpoints' Field

The 'Go To Breakpoints' field allows the user to enter up to 4 temporary breakpoints which, when reached, will cause emulation to stop. To enter a value, place the mouse cursor in this field and press the left mouse button. The user can then enter a hexadecimal beginning address in this field. A symbol can be entered by pressing the "Sym" button located next to the field and choosing the desired symbol. The "Scroll Box" button can be used to scroll through and select any previously entered symbols or addresses.

Once an address or symbol has been entered, pressing the "Go" button will begin emulation and the first occurrence of one of the defined breakpoints will cause emulation to stop at that point.

'Go From/To...' Window Buttons

The following buttons are available in this window:

Sym

The **Sym** button will pull up the Symbol Selection window. This window will allow a symbol name to be entered.

Go

The **Go** button will start emulation from the location entered in the 'Go From Address' field. If this field is blank, emulation will begin from the current program counter location.

Cancel

The **Cancel** button will cancel the 'Go From/To' operation and will close this window.

Help

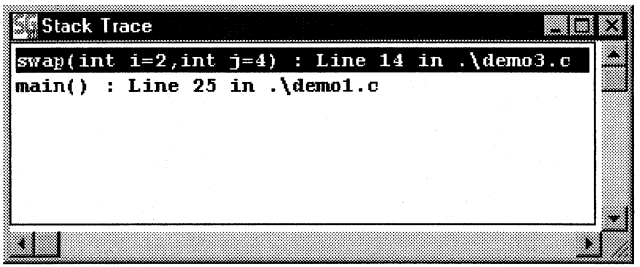
The **Help** button will provide help for this window.

Scroll Box

The **Scroll Box** button will allow the user to scroll through previously entered symbols or addresses.

Using The Stack Trace Window

The Stack Trace window shows the functions and local variables that are pushed onto the stack. The local variables passed and their values are displayed in hexadecimal. The Stack Trace window has the following format:



Opening The Stack Trace Window

To open the Stack Trace window, choose 'Display/Stack Trace'. An alternate method of opening the Stack Trace window is to use the accelerator key combination "Ctrl+K".

Stack Trace Window Contents

The Stack Trace window is automatically updated whenever the emulator is single-stepped, a breakpoint occurs, a stop command is executed or the emulator is reset. The information shown in this window is the true information that is currently located on the stack. This provides you a way to track what is put on the stack.

The information shown may be a combination of source line addresses and assembly addresses. The source line information will be in the format:

function (type var=value) : Line x in module

where:

function :	is the calling function
type:	is the variable type being passed
var:	is the variable name being passed
value:	is the value of the variable being passed
x:	is the line number of the calling module
module:	is the calling module

The assembly information will be in the format:

PC=X FP=Y

where:

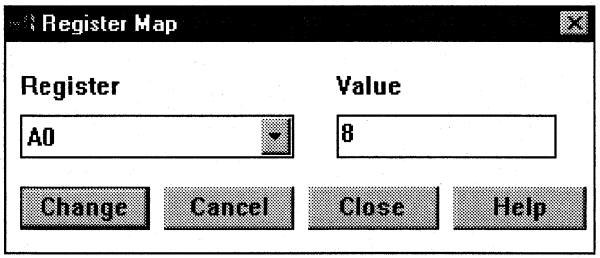
PC:	is the program counter
X :	is the value of the program counter
FP:	is the frame pointer
Y:	is the value of the frame pointer

An additional feature available from this window is the ability to call up a CodeView or assembly window by double clicking on a variable shown in the Stack Trace window or by pressing "Enter" while the desired Stack Trace entry is highlighted. Once this is done, the source or assembly module associated with that Stack Trace entry is shown in a CodeView or assembly window. For additional information, see Chapter 5 of this manual.

This version of SourceGate will display local variable values in this window for IAR Systems and Software Development Systems (SDS) compilers and any compiler that can produce Common Object File Format (COFF) or IEEE-695 output files. Future releases may provide this feature for other compilers.

Using The Register Map Window

This window allows the user to define the value used for a register when register variables are used. This value should match the value assigned to the register by the compiler. The Register Map window has the following format:



Once you have entered the desired value, press the **Change** button to store the change in SourceGate II. Consult the register variables section of your compiler documentation for information on what value to set for a particular register. Note that some compilers may not use register variables. In that case, this window will be meaningless.

Opening The Register Map Window

To open the Register Map window, choose 'Config/Register Map...'.

Register Map Window Buttons

The following buttons are available in this window:

Change

The **Change** button lets you change the mapping for a register name/register number pair. After making the desired changes, press the **Change** button to accept the changes.

Cancel

The **Cancel** button closes the Register Map window and discards all the changes you have made (via the **Change** button). The changes will NOT be saved in the <processor>.ini file.

Close

The **Close** button closes the Register Map window and saves all the changes you have made (via the **Change** button) in the <processor>.INI file.

Help

The **Help** button invokes help for this window.

C++ Class Browser/Sub Browser Windows

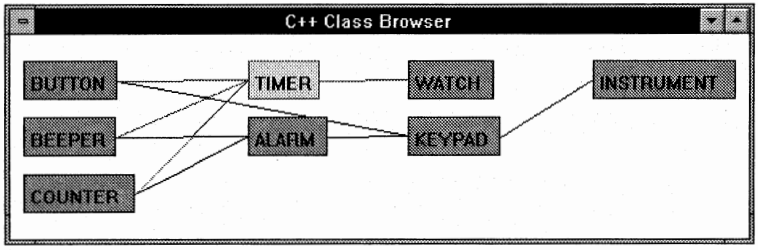
The C++ Class Browser and Sub Browser windows allow you to browse all C++ classes contained in the C++ source file. Each of these windows is described in the following sections.

Opening The C++ Class Browser/Sub Browser Windows

To open the C++ Class Browser window, select the "Display/C++ Class Browser" menu item. The Sub Browser window is opened from within a Browser window by double-clicking on a class object.

C++ Class Browser Window

This window allows you to browse all C++ classes contained in the source file. An example C++ Class Browser window is shown below:



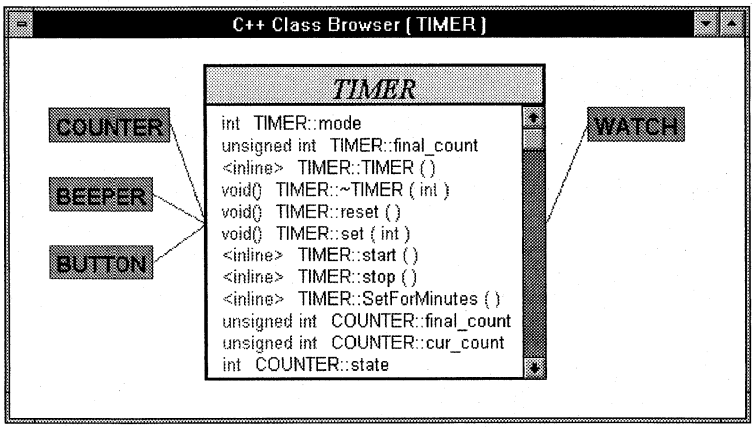
The Base (or Parent) classes are shown on the left side of the window. Derived (or Child) classes are shown to the right of the Base classes. Also shown are the lines of inheritance between the classes. These lines show you what classes were derived from the Base classes.

Class objects are shown in green. To select a class object, click on it once. Its color will then change from green to yellow. If you have selected a Base class, you will notice that the paths of inheritance to all classes derived from the Base class are shown in red. If you have selected a Derived class, you will notice that the paths of inheritance from all related Base classes will be shown in red.

When you double-click on a class object, a C++ Class Sub Browser Window is shown for that class object. This Sub Browser window shows paths of inheritance along with all elements of its own class and all elements of the classes that it was derived from (Base/Parent).

C++ Class Sub Browser Window

This window allows you to browse all C++ classes contained in the selected class object. An example C++ Class Sub Browser window is shown below:



This Sub Browser window shows paths of inheritance along with all elements of its own class and all elements of the classes that it was derived from (Base/Parent). It also shows any Base and Derived classes that are associated with the selected class. You can double-click on these other classes while in the sub browser window and this will pull up sub windows for those classes.

In addition, this window will show all data members and member functions. A member function will contain bracket () and a data member will not contain brackets. Some examples may help.

Data Member Example:

unsigned int Timer::final_count

In the above example, "unsigned int" is the type of element (shown in red), "Timer" is the class (shown in blue), and "final_count" is the variable that is a member of "Timer" (also shown in blue).

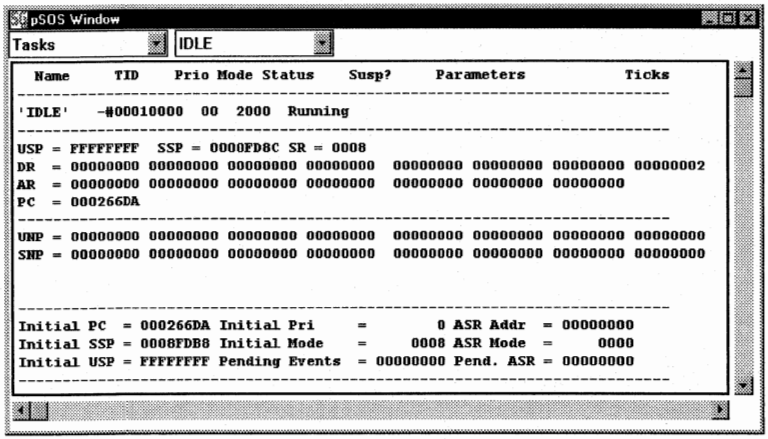
Member Function Example:

<inline> Timer::start(int)

In the above example, "<inline>" is the type of element (shown in red), "Timer" is the class (shown in blue), and "start" is the Member function that also shows the type of argument for the function ("int" in this example). Both the brackets and the argument types that they contain will be shown in red.

Using The pSOS Window

The pSOS window allows the user to view pSOS data. An example pSOS window is shown below:



The data shown in this window is determined by the two scroll boxes located at the top of the window. These scroll boxes allow the user to select the data to view. The left-most scroll box shows the main selection and the right-most scroll box shows any sub-options that are available for the main selection. The "Scroll Box" buttons show the options available for each scroll box.

The list of available scroll box selections and sub-option selections is listed below. Please consult your pSOS documentation for any questions about these available selections. If you have problems using the pSOS window, please contact HMI Technical Support.

The "Configuration" selection will allow the following configuration options to be selected:

Node
MPC
pSOS
pROBE
pHILE
pREPC
pNA
pSE
pMONT

The "Objects" selection will allow the following object types to be selected:

All (Shows all objects)
Tasks
Queues
Semaphores
Regions
Partitions

The "Partitions" selection will allow the following partition types to be selected:

All (Shows all partitions)

The "Queues" selection will allow the following queue types to be selected:

All (Shows all queues)

The "Regions" selection will allow the following region types to be selected:

All (Shows all regions)
RN#0

The "Semaphores" selection will allow the following semaphore types to be selected:

All (Shows all semaphores)
RDA1
WRA1
RXC1
TXC1

The "Tasks" selection will allow the following tasks types to be selected:

All (Shows all tasks)
IDLE
ROOT

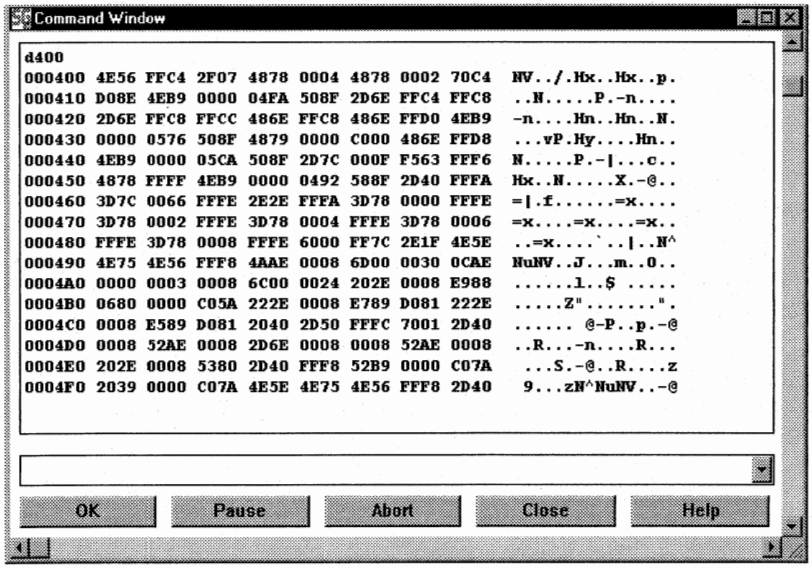
4 BMD Specific Features

SourceGate II, the Background Mode Debugger (BMD), and the background mode port that the SourceGate II/BMD system is controlling have a close relationship. When you send a command to the BMD, it must be translated into a command that the background mode port can understand.

Likewise, when the background mode processor sends information to the computer, the SourceGate II/BMD system must take this information and display it to you. This partnership makes up what we refer to as the "BMD Interface". In this chapter, we will attempt to explain the relationship between the background mode processor and the SourceGate II/BMD system, and we will show you how to set up the SourceGate II/BMD system to perform the emulation tasks you require. Each area of the "BMD Interface" will be explained in the sections that follow.

BMD Command Window

The BMD Command window is used to send commands to the BMD to control its operation. An example BMD Command window is shown below:



The following sections discuss BMD Command window specific topics.

Command Line Overrides

An override is an optional command parameter that allows the user to temporarily override the established global (default) parameter. Overrides are entered on the command line in square brackets or parenthesis. More than one override may be specified in the brackets and several may be entered immediately adjacent to each other without separating spaces or delimiters.

Command line overrides do not apply to all commands. They are represented in the command format as a square brackets enclosing the override codes that apply to the command. The following overrides are applicable to all BMDs:

S (Supervisor Mode)

Memory addresses given in the command are assumed to be references into either the Supervisor Data or Supervisor Program memory partition. Which one is used will be determined by other overrides or global parameters.

U (User Mode)

Memory addresses given in the command are assumed to be references into either the User Program or User Data memory partition. Which one is used will be determined by other overrides or global parameters.

P (Program Mode)

Memory addresses given in the command are assumed to be references into either the User Program or Supervisor Program memory partition. Which one is used will be determined by other overrides or global parameters.

D (Data Mode)

Memory addresses given in the command are assumed to be references into either the User Data or Supervisor Data memory partition. Which one is used will be determined by other overrides or global parameters.

L (Long Word Mode)

The optional data size of input parameter or output (display) values is selected to be "Long Word".

W (Word Mode)

The optional data size of input parameter or output (display) values is selected to be "Word".

B (Byte Mode)

The optional data size of input parameter or output (display) values is selected to be "Byte".

Available Command Window Commands

The following commands are available from the command line for all processor versions of the SourceGate II BMD:

A (In-Line Assembly)
BM (Byte Mode)
C (Compare Memory Block)
CM (Show Status Of Control Board Memory)
D (Dump Memory Block)
DM (Data Memory Mode)
E/EN (Enter/Change Memory)
EXP (Examine/Change Peripheral) Command
EXR (Examine/Change Peripheral Register) Command
F (Fill Memory Block)
G (Begin Real-Time Emulation)
HLD (Hex Listing Disabled)
HLE (Hex Listing Enabled)
I/IN (Input)
L (List Disassembled Code)
LM (Long Word Display Mode)
M (Move Memory Block)
MT/MTL (Test Block Of Memory)
O/ON (Output)
PM (Program Memory Mode)
RD (Read Object File Into Memory)
RP (Reset Processor, Continue Emulation)
RS (Reset Processor, Stop Emulation)
S_PMBA (Set Programmable Module Base Address)(CPU32/CPU32+ Only)
S_ROMWIDTH (Set ROM Width)(68360 Only)
SM (Supervisor Memory Mode)
SP (Stop Emulation)
SR/SRN (Search Block Of Memory)
SS (Single Step)
ST (Show Emulator Status)
UM (User Memory Mode)
VASM (View Assembly)
VER (Display Firmware Version)
W (Write Memory Contents To File)
WM (Word Display Mode)
X (Examine/Change Registers)

All of the preceding commands are described in detail in the sections that follow. Optional parameters are shown enclosed by parenthesis. Any differences between processor versions will be noted in the individual command descriptions.

A (In-Line Assembly) Command

This command allows assembly language mnemonics to be entered into memory starting at the specified address (In-Line Assembly). Once the Assembly command is entered, the address will be displayed with the cursor to the right. At this point, assembly language instructions can be entered. After each entry, the BMD displays the next address and prompts for the next instruction.

An error in the entered instruction will be flagged by displaying "-- Invalid Entry--" and re-displaying the current address.

Note: While this command is active, no other Command window commands are allowed to be entered. A period (.) followed by a carriage return terminates the entry sequence and the command.

The format for the Assemble command is:

A([SUPD]) Addr

where:

Addr:	is the hexadecimal starting address
SUPD:	are the available overrides

BM (Byte Mode) Command

This command sets the default for input or output values to "Byte Mode". While in Byte mode, commands such as D (Dump memory) will display values in byte mode instead of word mode or long word mode, which are the other choices (see WM and LM).

In addition, commands which require an input value will be affected by this command. For example, if the Fill command, F 8000 9000 33 is executed while in byte mode, each byte of memory from 8000H through 9000H will contain 33H. If this same command was executed with the BMD in word mode, then 0033H would be stored in each word location from 8000H to 9000H. If this same command was executed with the BMD in long word mode, then 00000033H would be stored in each long word location from 8000H to 9000H.

C (Compare Memory Block) Command

The Compare command will compare one block of memory to another. The first word in the first block will be compared to the first word in the second block, and so on. Any non-comparisons will be displayed on the screen showing the address in the first block with its data value and the address in the second block with its data value. If the two blocks match, then nothing will be displayed. **Note that you can also use the Memory window to compare memory.**

The format for the Compare command is:

C([SUPD]) Baddr1 Eaddr1 ([SUPD])Baddr2

where:

Baddr1: is the hexadecimal beginning address of the first compare block
Eaddr1: is the hexadecimal ending address of the first compare block
Baddr2: is the hexadecimal beginning address of the second compare block
SUPD: are the available overrides

The size of the first block sets the number of words to be compared. For each word in block one that does not match the corresponding word in block two, a line is displayed on the screen.

This display line has the format:

Addr1 Value1 Addr2 Value2

where:

Addr1:	is the address of the word in block one
Value1:	is the contents of addr1
Addr2:	is the address of the word in block two
Value2:	is the contents of addr2

CM (Show Status Of Control Board Memory) Command

Executing this command will show the version of the BMD that is being used.

D (Dump Memory Block) Command

The Dump Command will display the contents of memory in the specified address range showing the hexadecimal data with its ASCII equivalent on the screen. If no ending address is given, then one screen of data will be displayed and the command will terminate. If no starting address is specified then the last ending address (whether user entered or default) will be used as the starting address. Using this method of defaulting allows you to dump a display of memory from one starting address and get the next and subsequent pages of data dumped by simply typing the "D" and a carriage return. **Note that you can also use the Memory window to display the contents of memory.**

The Format for the Dump Command is:

D([SUPDLWB]) (Baddr) (Eaddr)

where:

Baddr:	is the hexadecimal beginning address of the block of memory to be displayed
Eaddr:	is the hexadecimal ending address of the block of memory to be displayed
SUPDLWB:	are the available overrides

DM (Data Memory Mode) Command

The execution of this command sets the default memory category to be Data Memory mode. Using the SM (Supervisor Mode) or UM (User Mode) commands further define the default memory mode.

E/EN (Enter/Change Memory) Command

The Enter Command is used to examine memory at the specified location with the option to change its contents. After executing the Enter Command, the selected address will appear on the screen with the current data. The data can be altered by entering new data and pressing a carriage return, or it can remain unchanged by just typing a carriage return. In either case the next address will be displayed with its data.

There is a variation of the Enter command (EN) that does not advance to the next address. Whether you enter a change and a carriage return or just a carriage return, it will display the value at the specified address only. This feature is very useful in monitoring a memory mapped I/O location.

Note: While these commands are active, no other Command window commands are allowed to be entered. A period (.) followed by a carriage return terminates the entry sequence and the command. Also note that you can use the Memory window to examine and change the contents of memory.

The Format for the Enter Command is:

E([SUPDLWB]) Addr
or
EN([SUPDLWB]) Addr

where:

Addr:	is the hexadecimal address of the data value to be displayed for possible substitution
SUPDLWB:	are the available overrides

EXP (Examine/Change Peripheral) Command

The EXP command changes the current default peripheral. The EXR command can then be used to examine/change a specific register within the default peripheral. Entering EXP by itself on the command line will show the current default peripheral.

The format of the EXP command is:

EXP per

Where:

per is the desired default peripheral whose registers are to be examined

EXR (Examine/Change Peripheral Register) Command

The EXR command allows a register in the current default peripheral to be examined/changed. The default peripheral can be selected by using the EXP command. Entering EXR by itself on the command line will show all registers associated with the current default peripheral.

The format of the EXR command is:

EXR reg

Where:

reg is the register peripheral to be examined or changed

F (Fill Memory Block) Command

The Fill Command is used to fill memory with a constant. The given data value will be written to each memory location within the specified range. The value may be a byte or a word in length, determined by the global parameters and overrides used. **Note that you can also use the Memory window to fill memory.**

The format for the Fill Command is:

F([SUPDLWB]) Baddr Eaddr Val

Where:

Baddr:	is the hexadecimal beginning address of the memory block to be filled
Eaddr:	is the hexadecimal ending address of the memory block to be filled
Val:	Constant value to be used to fill the memory block
SUPDLWB:	are the available overrides

G (Begin Real-Time Emulation) Command

This command is used to start real-time emulation. The options for this command are to run with temporary breakpoints as set in this command, or without a breakpoint. In either case, emulation will begin at the start address given or default to the current program counter. If a starting address is to be used it must be immediately adjacent to the G (otherwise it will be taken as a breakpoint address).

The CPU32 BMDs use software breakpoints only. The CPU32+ BMDs use software breakpoints only with the exception of the 68360 which allows for one hardware breakpoint to be set. The PowerPC BMD allows for up to four hardware breakpoints (address only) to be set. The ColdFire BMD allows for one hardware breakpoint (address only) to be set.

Note that, in all cases, when a breakpoint is set, SourceGate attempts to write a software breakpoint instruction into memory. If the memory where this write is attempted is not in RAM, one of the on-chip register-based hardware breakpoints is used (if the device has this capability). Therefore, if 128 breakpoints are set in RAM, none of the register-based breakpoints will be used. However, if the user attempts to set breakpoints in ROM, up to 4 breakpoints can be set in ROM using the on-chip register-based capabilities of the PowerPC devices and 1 breakpoint can be set in ROM using the 68360 and ColdFire devices.

If the CPU does not support register-based breakpoints or if the maximum number of available register-based breakpoints is exceeded, the message "Cannot Install Breakpoint Instruction At Location XXXXX" will be shown where "XXXXX" is the address in ROM where the breakpoint was attempted to be set.

The format for starting emulation with no temporary breakpoints is:

G(Saddr)

The format for setting temporary breakpoints is:

G(Saddr) Braddr1 (Braddr2 Braddr3 Braddr4)

Where:

Saddr:	is the hexadecimal address where execution (real-time emulation) will begin
Braddr1-4:	are hexadecimal addresses where execution (real-time emulation) will stop (or break) on op-code execution

HLD (Hex Listing Disabled) Command

Executing this command will disable the hex information from being displayed with the disassembly information during the L (List) command. To enable the hex information, execute the HLE (Hex Listing Enabled) command.

HLE (Hex Listing Enabled) Command

Executing this command will enable the hex information to be displayed with the disassembly information during the L (List) command. To disable the hex information, execute the HLD (Hex Listing Disabled) command.

I/IN (Input) Command

The I command is used to read and display a certain memory address. This function is very useful when monitoring memory mapped I/O locations. Each time a <cr> is entered, the memory location is read, shown, and incremented to point at the next location. If you wish to suppress the incrementing of the address every time a <cr> is pressed, the IN command should be used.

Note: While the IN command is active, no other Command window commands are allowed to be entered. A period (.) followed by a carriage return terminates the entry sequence and the command. Also note that you can use the Memory window to read and display memory.

The format for the Input command is:

I([SUPDLWB]) Addr

or

IN([SUPDLWB]) Addr

Where:

Addr:	is the hexadecimal address of the data value to be read
SUPDLWB:	are the available overrides

L (List Disassembled Code) Command

This command is used to disassemble the program memory data into assembly language mnemonics. All data and addresses are displayed in hexadecimal. If no ending address is specified, then one screen of disassembled mnemonics will be displayed and the command will terminate. If no beginning address is specified then the last ending address (whether entered or default) will be used as the beginning address. This allows you to disassemble code one display screen at a time by entering the command and an initial starting address for the first display and for the next and each subsequent page only the "L" and a carriage return need be entered. If a range is given, disassembled mnemonics will be displayed from the starting address to the ending address, scrolling the screen if necessary. To pause a scrolling screen enter CNTRL-S and, to resume scrolling a CNTRL-Q. This can also be accomplished by using the "pause" key to stop scrolling and <cr> to resume scrolling. To terminate the command while scrolling depress the space bar or enter a carriage return.

The format for the List Command is:

L([SUPD]) (Baddr) (Eaddr)

Where:

Baddr:	is the hexadecimal beginning address of the memory data to be disassembled
Eaddr:	is the hexadecimal ending address of the memory to be disassembled
SUPD:	are the available overrides

LM (Long Word Display Mode) Command

This command sets the default for input or output values to "Long Word Mode". While in Long Word mode, commands such as D (Dump memory) will display values in Long Word mode instead of word mode or byte mode, which are the other choices (see WM and BM).

In addition, commands which require an input value will be affected by this command. For example, if the Fill command, F 8000 9000 33 is executed while in Long Word mode, each long word of memory from 8000H through 9000H will contain 00000033H. If this same command was executed with the BMD in word mode, then 0033H would be stored in each word location from 8000H to 9000H. If this same command was executed with the BMD in byte mode, then 33H would be stored in each byte location from 8000H to 9000H.

M (Move Memory Block) Command

The Move Command is used to copy the contents of one block to the contents of another block (non- destructive move). Block one is defined with its starting and ending addresses being the first two address fields displayed, and block two is defined with its starting address being the third address field. The contents of the first memory location of block one is copied to the first memory location of block two, and so on. The next prompt character is not displayed until the move is complete. **Note that you can also use the Memory window to move memory.**

The format of the Move Command is:

M([SUPDR]) Baddr1 Eaddr1 ([SUPD])Baddr2

where:

Baddr1: is the hexadecimal beginning address of the data "Source" block
Eaddr1: is the hexadecimal ending address of the data "Source" block
Baddr2: is the hexadecimal beginning address of the data "Destination" block . The size of the "Source" block defines the number of bytes to be copied into the "Destination" block.
SUPD: are the available overrides

MT/MTL (Test Block Of Memory) Command

This command can be used to test memory (RAM) within the given address range.

The short memory test (MT) writes, reads and compares two separate bit patterns for each byte within the range. The bit patterns are 55H and AAH, thus cycling all bits of the byte through both one and zero.

The long memory test (MTL) is a walking bit pattern. This test writes, reads and compares the byte while having only one bit in the byte on at a time. This is done by using the Hexadecimal values 1, 2, 4, 8, 10, 20, 40, 80 for each byte within the range.

In either test, any read not comparing with what was written into a location will cause the address, data written and data read to be displayed on the next line of the screen. Both of these tests are non- destructive and can be performed without corrupting data currently in memory. A <cr> will terminate this test at any time.

The format for the Memory Test command is:

MT([SUPD]) Baddr Eaddr

The format for the long Memory Test command is:

MTL([SUPD]) Baddr Eaddr

Where:

Baddr: is the hexadecimal beginning address of the memory block to be tested.
Eaddr: is the hexadecimal ending address of the memory block to be tested.

O/ON (Output) Command

The O command is used to write to a certain memory address. This function is very useful when testing memory mapped I/O locations. Each time a <cr> is entered, the desired value is written to the displayed address and the address is incremented. The ON command will not increment the address to be written but will write the same address each time a <cr> is entered.

Note: While the ON command is active, no other Command window commands are allowed to be entered. A period (.) followed by a carriage return terminates the entry sequence and the command. Also note that you can use the Memory window to output the contents of memory.

The format for the Output command is:

O([SUPDLWB]) Addr,Val

or

ON([SUPDLWB]) Addr,Val

Where:

Addr:	is the hexadecimal address to be written with the defined value
Val:	is the hexadecimal value to be written
SUPDLWB:	are the available overrides

PM (Program Memory Mode) Command

The execution of this command sets the default memory category to be Program Memory mode. Using the SM (Supervisor Mode) or UM (User Mode) commands further define the default memory mode.

RD (Read Object File Into Memory) Command

This command is used to transfer a hexadecimal file from a host computer to the target system memory. The file name can be specified with this command. When downloading a file using the SourceGate II BMD, the ASCII file is converted to binary before being transferred to the target system. This essentially doubles the speed of the transfer over the normal ASCII transfer. **Note that you can also use the 'File/Download Object File...' menu item to read a file.**

The file formats accepted by the SourceGate software are Motorola S-Record, Intel Hex, Texhex, HMI Binary, Extended Tek Hex, Microtec Research Absolute, Common Object File Format (COFF), IAR Systems, Software Development Systems (SDS), IEEE-695, ELF/DWARF, and Hitachi.

Once the Read Command has been executed, a File Load Status window will appear which will show the status of the read operation. The read operation can be aborted by pressing the "Cancel" button located in the File Load Status window.

When reading a file from the main RS232 port, the format is:

RD([SUPD])(Offset),Filename,(Flags)

Where:

SUPD:	are the available overrides
Offset:	is a hexadecimal value representing the address offset
Filename	is a file name, valid to the host computer, that designates the file to be down loaded to the BMD. If no extension is specified, the default defined in the "Config/Preferences" window will be used.
Flags:	are options to be used for each read. The options are defined as follows:
A:	Download hex files to the target system using ASCII file transfer
B:	Download hex files to the target system using binary file transfer
C:	Concatenate any symbols found reading a hex file with those which already exist in the internal symbol table (i.e. do not clear the symbol table on a read command)
D:	Use only the digits presented when looking up a symbol by value for display (Opposite of "Z")

H: Upload/Download only the hex data records present in a hex file. Ignore all symbol records.

S: Upload/Download only the symbol records of a hex file. Ignore all hex data records.

M: Toggle between reading "S" record files as Motorola and Hitachi formats. Default is Motorola for Motorola microprocessors.

Z: Pad a number with leading zeros when doing a lookup by value for symbol display (Opposite of "D").

The default option control flag settings are "BZ".

The 'RD' command opens the named format file and reads its symbols into the internal SourceGate symbol table, and its hex data into the BMD's memory. If no extension is specified, the default defined in the "Config/Preferences" window will be used.

If the optional read bias is given, this offset value will be added to the addresses of all bytes read from the file, before these bytes are stored in memory. This offset is a hexadecimal number. If the filename is not given, the 'RD' command will use the filename given in the last 'RD' command.

Note: The default file extension used on read (load) and write (dump) operations can be changed by modifying the .SG2 file for your BMD. For read operations, the "DefaultLoadExtension" definition can be changed to reflect the desired extension. For write operations, change "DefaultDumpExtension".

RP (Reset Processor, Continue Emulation) Command

The RP command allows you to reset the processor without stopping emulation.

Note: The RP command can only be executed while emulation is in progress.

RS (Reset Processor, Stop Emulation) Command

This command will issue a reset to the processor. All registers will be updated to the value which they contain upon a processor reset.

S_PMBA (Set Programmable Module Base Address) Command (CPU32/CPU32+ Only)

The command S_PMBA (which may be entered using the command window or in a command script) allows the programmable module base address to be viewed and set. The format of the S_PMBA command is:

S_PMBA [<New Base>]

where:

New Base: is the desired location of the module base address

If no base address is specified, the current value is displayed. Otherwise, the specified address is validated, and the SourceGate II BMD sets the new address as follows:

68340/349/360:
The MBAR register is programmed to move the peripheral modules to the specified base address. The EXR command reads the MBAR register on every access to determine the register addresses.

68331/332:
The base address is recorded, and the EXR command uses this recorded address to determine register addresses.

Note: The SourceGate II BMD does not attempt to program the base address or determine the base address dynamically because the bit which is used to select the base address is itself relocatable, thus making it impossible for the SourceGate II BMD to determine the base address reliably. Therefore, for these processors, the S_PMBA command is used only to notify the SourceGate II BMD of the base address to use.

Some examples may help clarify how this command functions:

To determine the current base address:

```
S_PMBA
Peripheral Module Base Address = FFFFF000
```

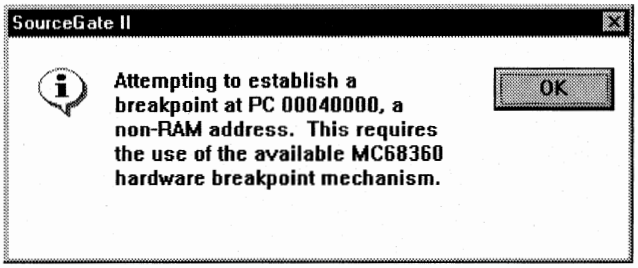
To set the base address to a new value:

```
S_PMBA FFF000
Peripheral Module Base Address = 00FFF000
```

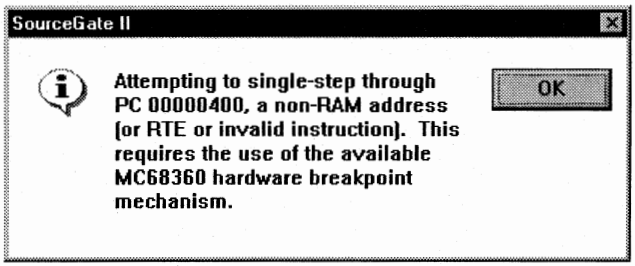
S_ROMWIDTH (Set ROM Width) Command (68360 Only)

The 68360 processor contains registers which may be used to establish a breakpoint in ROM. SourceGate II BMD for the 68360 uses these registers when the user tries to set a breakpoint in, or single-step through, ROM.

These registers do not work in all cases. If the PC lies in 16 bit-wide or 8 bit-wide memory, programming the registers works only if the Break Address Register (BKAR) is set to a 32 bit-aligned address. If it is set to a 16 bit-aligned address, the processor does not stop executing. Thus, when the user first attempts to set a breakpoint in ROM, SourceGate II displays the following series of dialogs:

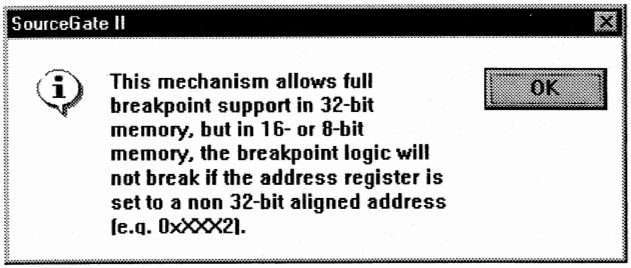

SourceGate II
Attempting to establish a breakpoint at PC 00040000, a non-RAM address. This requires the use of the available MC68360 hardware breakpoint mechanism.
OK

OR

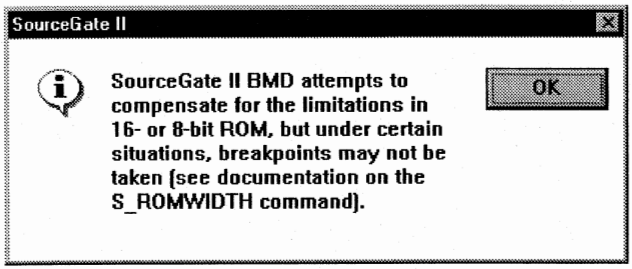

SourceGate II
Attempting to single-step through PC 00000400, a non-RAM address (or RTE or invalid instruction). This requires the use of the available MC68360 hardware breakpoint mechanism.
OK

The dialog shown will depend on the operation being performed (single-stepping or setting a breakpoint).

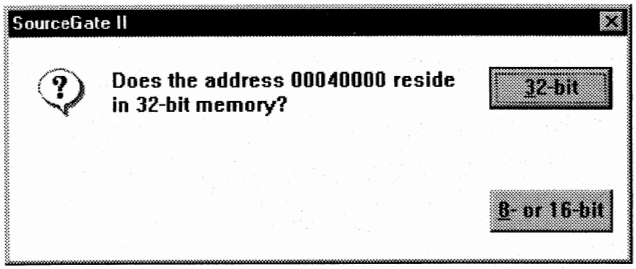
THEN


SourceGate II
This mechanism allows full breakpoint support in 32-bit memory, but in 16- or 8-bit memory, the breakpoint logic will not break if the address register is set to a non 32-bit aligned address (e.g. 0xXXX2).
OK

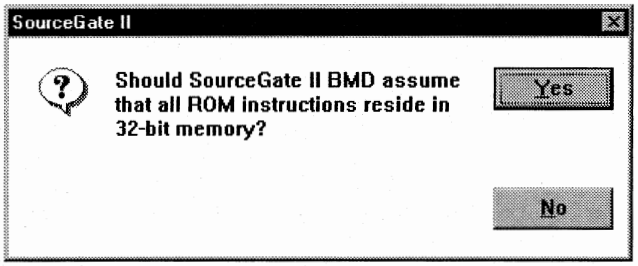
THEN


SourceGate II
SourceGate II BMD attempts to compensate for the limitations in 16- or 8-bit ROM, but under certain situations, breakpoints may not be taken (see documentation on the S_ROMWIDTH command).
OK

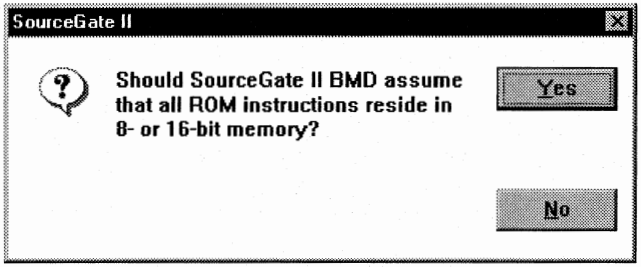
THEN



THEN

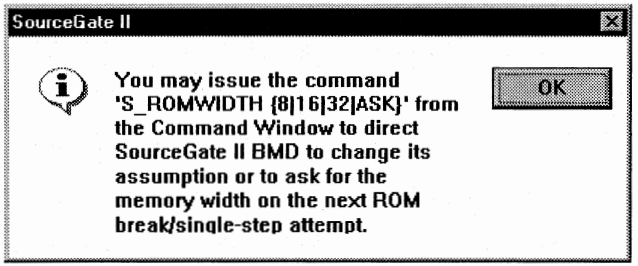


OR



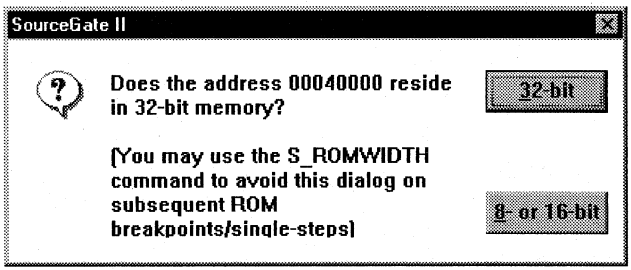
The dialog shown will depend on the answer given for the previous dialog.

THEN


SourceGate II
You may issue the command 'S_ROMWIDTH {8|16|32|ASK}' from the Command Window to direct SourceGate II BMD to change its assumption or to ask for the memory width on the next ROM break/single-step attempt.
OK

On subsequent attempts to set breakpoints, if the user has specified (via the above dialog boxes or by using the S_ROMWIDTH command) that SourceGate II should always assume a specified size, SourceGate II does assume the specified size, and doesn't ask any questions.

Otherwise, the next time the user tries to establish a breakpoint in ROM or single-step while in ROM, they will get this dialog:


SourceGate II
Does the address 00040000 reside in 32-bit memory?
32-bit
(You may use the S_ROMWIDTH command to avoid this dialog on subsequent ROM breakpoints/single-steps)
8- or 16-bit

At this point, if the user specifies a 32-bit ROM, then the breakpoint address register is simply set to the address specified. **Note: Make sure that, if "32-bit ROM" is selected, the system actually does have a 32-bit ROM. If "32-bit ROM" is selected but the breakpoint address is 16-bit aligned, the breakpoint will never occur, even if the instruction is executed.**

If the user specifies a 16-bit ROM, then SourceGate II rounds up the address to the next 32-bit boundary. This usually means that the breakpoint will occur a little later than normal. However, if the instruction where the breakpoint is specified is a 16-bit branch instruction which is 16-bit aligned, then the breakpoint will not occur when the instruction is executed.

An example may help:

```
000004D8 MOVE.L (0008,A6),D7
000004DC TST.L D7
000004DE BLT 00000502
000004E0 MOVEQ #03,D0
000004E2 CMP.L D0,D7
000004E4 BGE 00000502
000004E6 MOVE.L D7,D0
000004E8 LSL.L #4,D0
```

In the above example, the instructions at locations 4DE, 4E2, and 4E6 are 16-bit instructions that are 16-bit aligned. If the user attempts to set a breakpoint at one of these addresses and they have indicated that the address lies in 8 or 16-bit ROM, SourceGate II will round the address up and set a breakpoint at 4E0, 4E4, or 4E8. In the case of the instructions at 4E2 and 4E6, this is not a big problem because the breakpoint will occur just one instruction later. However, in the case of the branch instruction at 4DE, SourceGate II will set the breakpoint at address 4E0 and, if the branch is taken, the breakpoint will never be occur.

The format of the S_ROMWIDTH command is:

S_ROMWIDTH {8 | 16 | 32 | ASK}

Some examples follow:

S_ROMWIDTH 8:	assume 8 or 16-bit ROM for all breakpoints/single-steps
S_ROMWIDTH 16:	assume 8 or 16-bit ROM for all breakpoints/single-steps
S_ROMWIDTH 32:	assume 32-bit ROM for all breakpoints/single-steps
S_ROMWIDTH ASK:	ask the user for the ROM width next time they try to set a breakpoint or single-step in ROM.

SM (Supervisor Memory Mode) Command

The execution of this command sets the default memory category to be Supervisor Memory mode. Using the PM (Program Mode) or DM (Data Mode) commands further define the default memory mode.

SP (Stop Emulation) Command

This command is used to stop emulation in order to regain control of the processor. Upon entering this command, emulation will break, and the message "PROCESSOR IS IDLING" will be displayed and the register window will be updated with the current register values.

SR/SRN (Search Block Of Memory) Command

This command can be used to search memory between two addresses for a particular data pattern. When a match is found, the address is displayed. To search for the next occurrence, hit the Carriage Return key. To terminate this command, type a period. If no match is found, then nothing is displayed.

There is also a Search Not option available. In this mode the value is searched for, but the command displays the address of all memory locations in the given range that do not match the search value. This is useful when checking a block of memory that should contain a constant value, such as zero or the value of a previous Fill command.

Note that you can also use the Memory window to search the contents of memory.

The format for the Search command is:

SR([SUPDLWB]) Baddr Eaddr Val

The format for the Search Not command is:

SRN([SUPDLWB]) Baddr Eaddr Sval

where:

Baddr:	is the hexadecimal beginning address of the block of memory to be searched
Eaddr:	is the hexadecimal ending address of the block of memory to be searched
Val:	is the data value to be searched for and will be 1 byte or word value or a 1-20 character ASCII string enclosed in single quotes.
Sval:	is the data value to be searched for and will be a byte value, a word value or one or two ASCII characters enclosed in quotes.
SUPDLWB:	are the available overrides

SS (Single Step) Command

This command is used to execute one instruction at a time. Each time a step is performed, the address and instruction to be executed are displayed along with the register values from the previous instruction executed. Therefore, multiple steps will display a history of program flow.

There are four options for operating the Single Step Command. The first option is to single step every time a carriage return is entered. **To terminate this mode, type a period.** The second option is to single step a specified number of times. The third option is to single step continuously. The fourth option allows you to single step "Until" some condition. **To terminate the second, third, and fourth options, a carriage return should be entered.**

The format of the single step command is as follows:

SS[<count>][X] [Until clause] [Q/Q1/Q2] [Skip]

Where:

SS:	Steps once each time a <cr> is entered. A period (.) aborts stepping
SS<count>:	Steps <count> times without waiting for a <cr>. A <cr> aborts stepping
SSX:	Steps continuously until a <cr> is entered
SS Until clause:	Steps until the condition in the clause is true. A <cr> aborts stepping

The "Quiet" option is invoked by appending a "Q" to the end of the command. In this mode, only the first and last step are displayed (e.g. SSX Q will step automatically until a <cr> is entered showing the first and last instructions stepped). If "Q1" is used, only the last step is shown. If "Q2" is used, all steps are shown in the form "Addr Instruction" where "Addr" is the value of the PC and "Instruction" is the assembly language instruction pointed to by that PC value. In this mode, all register values except the PC are suppressed.

The optional "SKIP" parameter causes all subroutine calls to be shown as a single step. The subroutine called will be executed but the information shown while stepping will not include the instructions in the subroutine, only the call to it. If the subroutine called, in turn, calls other subroutines, or if the subroutine called contains a large number of instructions, there may be a noticeable wait until the program returns to the main program.

The Until clause is defined as follows:

```
OPERAND1 OPERATION OPERAND2

Where OPERAND1 can be:

Reg <reg> - Register name
Long at <addr> - Long word value at address <addr>
Word at <addr> - Word value at address <addr>
Byte at <addr> - Byte value at address <addr>

OPERAND2 can be:

Reg <reg> - Register name
#<value> - Immediate value (size determined by OPERAND 1)
<addr> - Value at <addr> (size determined by OPERAND 1)

OPERATION can be:

 = - OPERAND1 equal to OPERAND2
== - OPERAND1 equal to OPERAND2
!= - OPERAND1 not equal to OPERAND2
<> - OPERAND1 not equal to OPERAND2
>< - OPERAND1 not equal to OPERAND2
 > - OPERAND1 greater than OPERAND2
 < - OPERAND1 less than OPERAND2
>= - OPERAND1 greater than or equal to OPERAND2
=> - OPERAND1 greater than or equal to OPERAND2
<= - OPERAND1 less than or equal to OPERAND2
=< - OPERAND1 less than or equal to OPERAND2
```

Example:

SS Until word at 5000 = #1234

Would stop stepping when 1234H appears at address 5000H

It should be noted that this command takes into account the way data is stored in memory based on the microprocessor manufacturer. If, for example, you wished to stop on a word value of 1234H being in location 5000H, the BMD will look for a 1234H at location 5000H for Motorola processors and 3412H at location 5000H for Intel, Hitachi, and Zilog processors.

ST (Show Emulator Status) Command

Executing the Status command will display the current status of the BMD.

UM (User Memory Mode) Command

The execution of this command sets the default memory category to be User Memory mode. Using the PM (Program Mode) or DM (Data Mode) commands further define the default memory mode.

VASM (View Assembly) Command

This command will open an assembly Code window beginning at the specified address. The format of the VASM command is:

VASM <addr>

Where:

addr: is the beginning address of the VASM window

If no address is entered, the VASM window will begin with the address of the current program counter.

See Chapter 3 -*Viewing Assembly Code In CodeView Windows* for additional information on using the VASM window.

VER (Display Firmware Version) Command

This command displays the current version and release date of the BMD firmware.

W (Write Memory Contents To File) Command

This command is used to transfer files from the target system memory to a host computer. The size of the block being transferred is specified by the two address fields activated by this command. When uploading a file using the SourceGate II BMD, the block will be written as a Motorola S-Record file with symbol information added to the beginning of the file. This symbol information is in a format that is easily read by HMI software. If no extension is specified, the default defined in the "Config/Preferences" window will be used. This symbol/hex file is written to disk on the host computer under the specified filename. **Note that you can also use the 'File/Upload Object File...' menu item to write a file.**

Once the Write Command has been executed, a File Load Status window will appear which will show the status of the write operation. The write operation can be aborted by pressing the "Cancel" button located in the File Load Status window.

The format for writing to the main RS232 port is:

W([SUPD]) Baddr Eaddr Filename (Flags)

Where:

SUPD:	are the available overrides
Baddr:	is the hexadecimal beginning address of code to be written
Eaddr:	is the hexadecimal ending address of code to be written
Filename:	is a filename, valid to the host computer, that designates the file to be uploaded (written) to the host computer.If no extension is specified, the default defined in the "Config/Preferences" window will be used.
Flags:	are options to be used for each write The options are defined as follows:
H:	Upload/Download only the hex data records present in a hex file. Ignore all symbol records.
S:	Upload/Download only the symbol records of a hex file. Ignore all hex data records.

The 'W' command opens the given file for writing, and writes the contents of the SourceGate internal symbol table into this file, along with the data bytes in memory, from Baddr through Eaddr. 'Baddr' and 'Eaddr' are hexadecimal numbers (or symbols). **Note: The default file extension used on read (load) and write (dump) operations can be changed by modifying the .SG2 file for your BMD. For read operations, the "DefaultLoadExtension" definition can be changed to reflect the desired extension. For write operations, change "DefaultDumpExtension".**

WM (Word Display Mode) Command

This command sets the default for input or output values to "Word Mode". While in Word mode, commands such as D (Dump memory) will display values in Word mode instead of long word mode or byte mode, which are the other choices (see LM and BM).

In addition, commands which require an input value will be affected by this command. For example, if the Fill command, F 8000 9000 33 is executed while in Word mode, each word of memory from 8000H through 9000H will contain 0033H. If this same command was executed with the BMD in long word mode, then 00000033H would be stored in each long word location from 8000H to 9000H. If this same command was executed with the BMD in byte mode, then 33H would be stored in each byte location from 8000H to 9000H.

X (Examine/Change Registers) Command

This command allows you to examine and modify the registers and flags (status register). There are three options to the examine command. The first option is to display all registers and flags. The second option is to display a specific register and allow it to be modified. The last option is to allow the flags to be modified by typing in the condition code status. **Note that you can also use the Register window to display and change the register contents.**

The format for displaying the registers and condition codes is:

X

The format for displaying a specific register and allowing it to be modified is:

XReg

Where:

Reg: is any processor register

When changing a register using the "Xreg" command, you will be shown the current value of the register. To change it, simply type in the new value and then press the "ENTER" key. The register will then contain the new value.

The BMD 'Setup' Menu

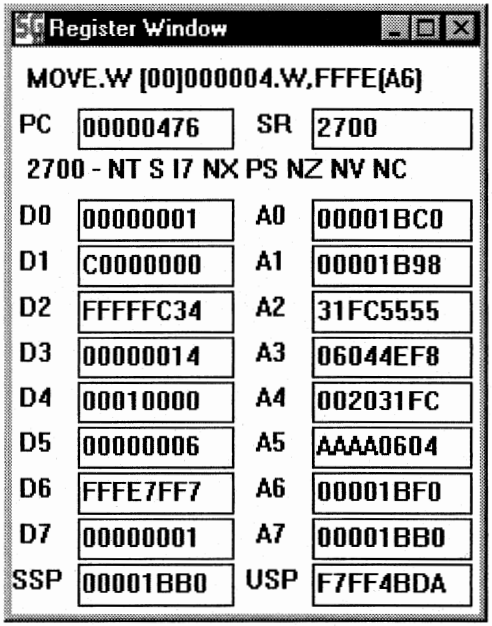
The SourceGate II 'Setup' menu is used to set up events, breakpoints, trigger points, and other system specific operations. The information shown in this window varies depending on the HMI system that you are using. **For the BMDs, all of these menu items are grayed out and are unavailable to the user.**

Using The Register Window

The Register window shows the contents of the processor's registers including the status of the flags and the current disassembly mnemonic being pointed to by the program counter register. The register values are always accurate whenever emulation is not in progress, and are the last known value whenever emulation is in progress.

Registers can be altered in the register display window by selecting the desired register data field and then and entering a new value.

The Register window is automatically updated whenever the BMD is single-stepped, a breakpoint occurs, a stop command is executed, or the BMD is reset. The registers shown in the Register window are dependent on the background mode processor being used. An example Register window is shown below:



Opening The Register Window

To open the Register window, choose 'Display/Registers'. An alternate method of opening the Register window is to use the accelerator key combination Ctrl+R.

Modifying Register Values

To modify a register, move the insertion bar to the register to be changed and then press the left mouse button. You can then enter the desired information beginning at the insertion point. To delete the old contents of a register and add a completely new value, click and drag to highlight the entire contents of the register and then enter the new value.

Available Registers

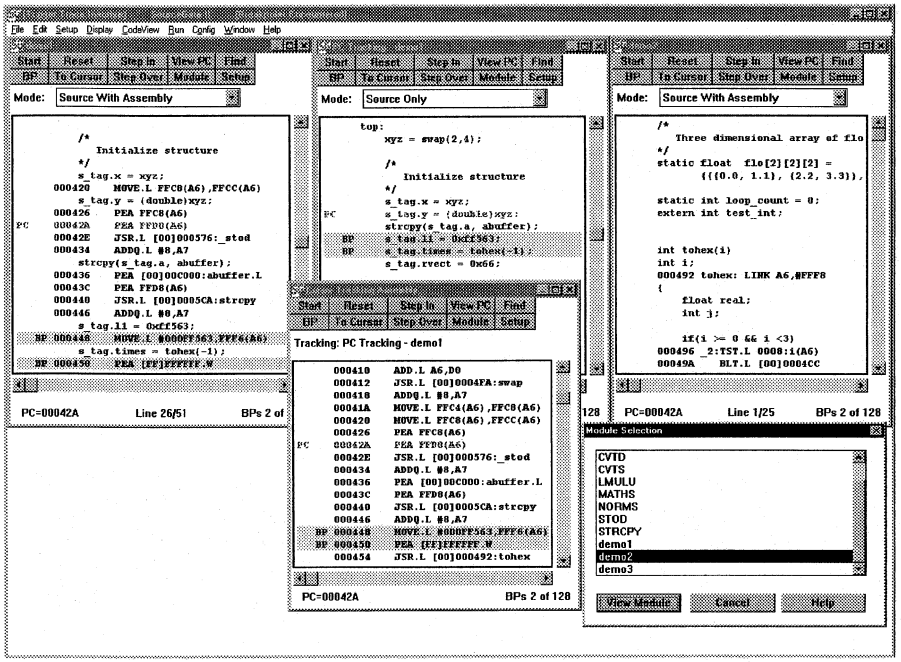
Consult the on-line help for a description of the registers specific to your BMD that are shown in the Register window. If you don't see a register of interest in the Register window, try the "EXP" and "EXR" commands from the Command window. These commands show other registers unique to the particular processor of interest.

5 The CodeView Interface

What Is The CodeView Interface?

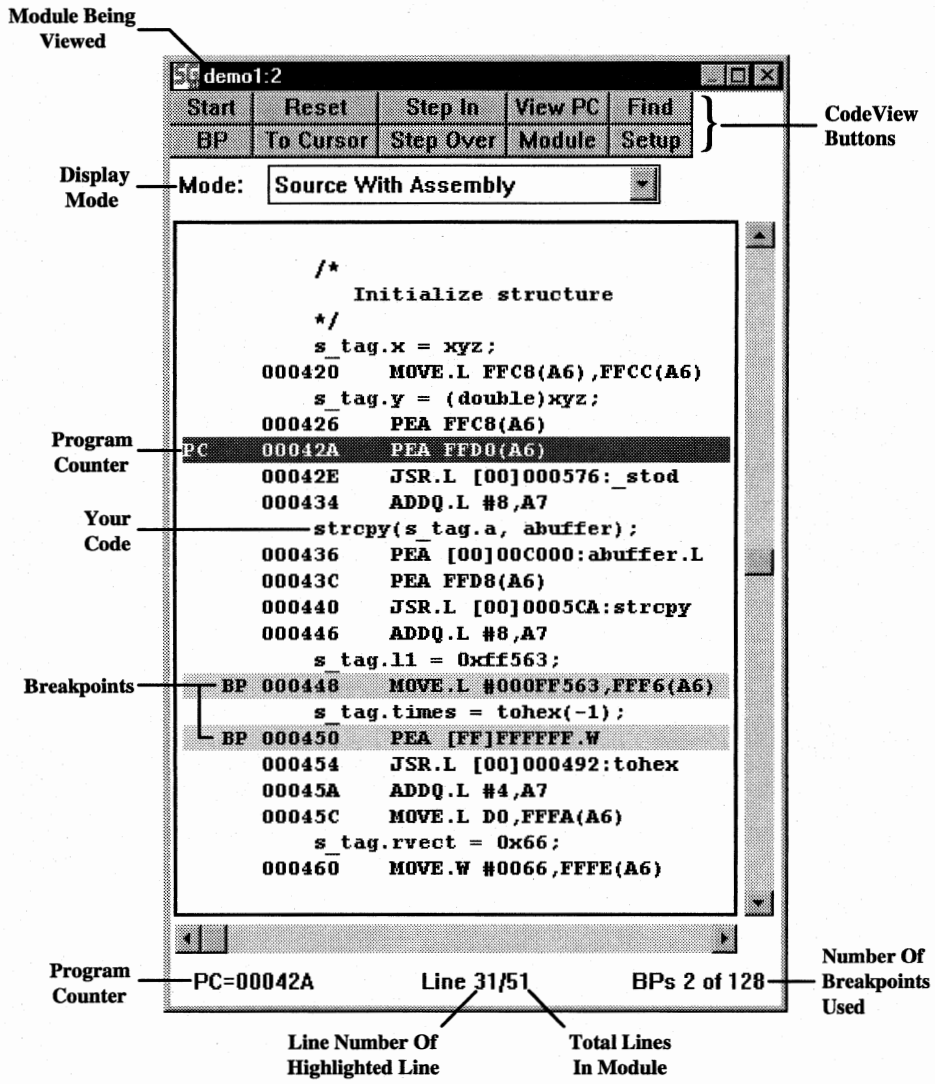
The CodeView interface provides you with the ability to control your HMI Development System directly from your source code file. Unlimited CodeView windows can be opened at any one time and can be set to display source, assembly, or a combination of source and assembly. Hardware or software breakpoints can be set in these CodeView windows. Single-stepping can be performed from these windows with the ability to step into or over functions. Preferences can be set to customize CodeView display parameters and functionality. With this wealth of features, it is easy to see why the SourceGate II CodeView interface will be invaluable to you in your debugging efforts.

An example CodeView window session is shown below:



CodeView Window Fields

The following fields are found in CodeView windows:



An explanation of each of these fields is given in the following sections.

Module Being Viewed
The top border of the CodeView window will show the name of the module being viewed. If multiple CodeView windows are opened for a particular module, the duplicate windows are designated in the top window border with the label "module:n" where "module" is the module that is being viewed and "n" is the nth window that has been opened for that module. In the example window shown, "demo1:2" designates this window as being the 2nd CodeView window opened for the module "demo1". For additional information on how to load files into the CodeView windows, see ***Loading Source Code Into CodeView Windows***.

CodeView Buttons
The buttons located in the CodeView window perform a number of different functions. For additional information, see ***CodeView Window Buttons***.

Display Mode
To change to one of the other modes of operation, simply click on the "Mode:" field and then click on the desired display mode. The CodeView window will then be updated in the newly selected mode. For additional information on changing the display mode, see ***Changing The Display Mode In CodeView Windows***. Note that the "Mode:" field is not shown in VASM type CodeView windows but is replaced by a "Range:" field. For additional information about this field, see ***Viewing Assembly Code In CodeView Windows***.

Your Code
This is the actual source code or assembly language file information. For additional information on how to load files into the CodeView windows, see ***Loading Source Code Into CodeView Windows***.

Program Counter (PC)
If the program counter (PC) is located in the CodeView window, it will be shown with a "PC" label next to the line pointed to by the PC. The current PC value is always shown in the bottom-left-hand corner of all open CodeView windows.

Breakpoints (BP)
If any breakpoints are set in a CodeView window, they will be shown with a "BP" label next to the line that is set to be a breakpoint. For additional information on how to set breakpoints in a CodeView window, see ***Setting Breakpoints In CodeView Windows***.

Line Number Of Highlighted Line/Total Lines In Module

The line number of where the highlight bar is located in a CodeView window is shown in this field along with the total number of available lines of code in the module. In the example window shown, the highlight bar is located on line 31 and there are 51 total lines in the module "demo1". Note that this field is not shown for VASM type CodeView windows.

Number Of Breakpoints Used

This field shows the number of breakpoints that have been set by pressing the "BP" button along with the total number of breakpoints available. For additional information on how to set breakpoints in a CodeView window, see ***Setting Breakpoints In CodeView Windows***.

CodeView Window Buttons

The buttons in the CodeView window allow the user to perform certain actions. Each of the available buttons are listed in the sections that follow.

Start

The **Start** button will begin emulation from the current program counter location. For additional information, see ***Starting Emulation From CodeView Windows***.

Note: When using the CodeView window, emulation must be started by pressing the "Start" button, F5 key, or the "CodeView/Start" menu item. Starting emulation via other methods ("G" command, "Run/Go" menu item, "Go" button) will NOT activate the CodeView window breakpoints and other CodeView specific functions.

BP

The **BP** button will toggle the breakpoint function for the highlighted CodeView line. When a breakpoint is active for a line, 'BP' will appear to the left of the line. Up to 128 breakpoints can be set to be active at any one time. Breakpoints can be set in different CodeView windows if desired. The "BPs x of 128" field located in the bottom right-hand corner of each CodeView window is constantly updated to reflect the number of breakpoints that have been set in all open CodeView windows.

It should be noted that, if the user attempts to set a breakpoint on a source line that does not have a compiler generated line number symbol associated with it (meaning the line generated no code), the message "No Assembly For Selected Line" will be generated. Breakpoints can only be set on source lines that generate assembly code or actual assembly lines themselves.

For additional information, see ***Setting Breakpoints In CodeView Windows***.

Reset

The **Reset** button will reset the CPU. All registers will reflect the value that they normally have after a processor reset. Certain HMI in-circuit emulators have Memory Configuration window values that are programmed after a processor reset. After the **Reset** button is pressed, these registers will contain the values defined in this window.

To Cursor

The **To Cursor** button will start emulation and run until the highlighted line is reached. For additional information, see ***Setting Breakpoints In CodeView Windows.***

Step In
The **Step In** button will step one line of code each time it is pressed, stepping into any calls to subroutines or other functions. If the user wishes to step over subroutines or function calls, the **Step Over** button should be used. For additional information, see ***Single-Stepping In CodeView Windows***.

Step Over
The **Step Over** button will step one line of code each time it is pressed, stepping over any calls to subroutines or other functions. If the user wishes to step into subroutines or function calls, the **Step In** button should be used. For additional information, see ***Single-Stepping In CodeView Windows***.

View PC
The **View PC** button will place the highlighted line on the line of code pointed to by the program counter. If the program counter is not located in the current active module, a new CodeView window (or VASM window if the PC line has no source code associated with it) will be opened and the highlighted line will be placed on the line containing the program counter.

The user may wish to open a PC Tracking window in order to track the current PC location. For additional information on how to do this, see ***Using The PC Tracking Window***.

Module
The **Module** button will invoke the Module Selection window. The user can then choose a module which will be loaded into a CodeView window. For additional information, see ***Loading Source Code Into CodeView Windows***.

Find
The **Find** button will invoke the CodeView Find window. This window can be used to search for strings in the active CodeView window. For additional information, see ***Searching The CodeView Window***.

Setup
The Setup button will invoke the CodeView Setup window for the CodeView window from which the button was pressed. This window allows the user to customize the look and behavior of each individual CodeView window. For additional information, see ***Setting CodeView Window Preferences.***

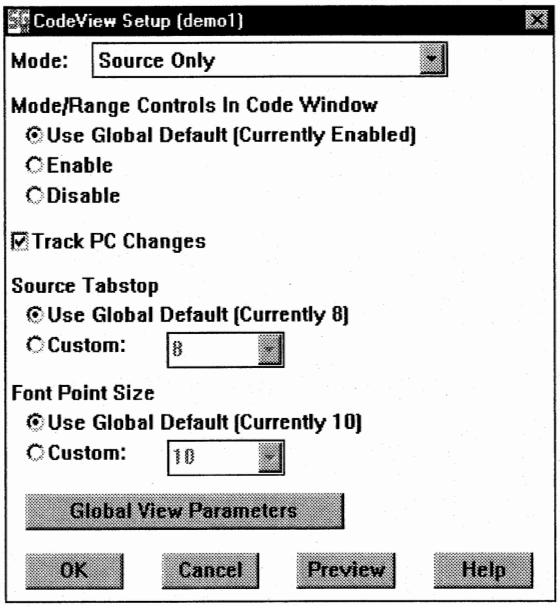
Setting CodeView Window Preferences

The setting of CodeView window preferences is accomplished through the use of the CodeView Setup windows which allow the user to customize how the CodeView windows look and behave. To change the CodeView window preferences, press the "Setup" button located in the CodeView window or select the "Config/Preferences/CodeView View..." or "Config/Preferences/CodeView Function..." menu items.

There are four separate windows that define CodeView preferences. Each of these windows will be discussed in the sections that follow.

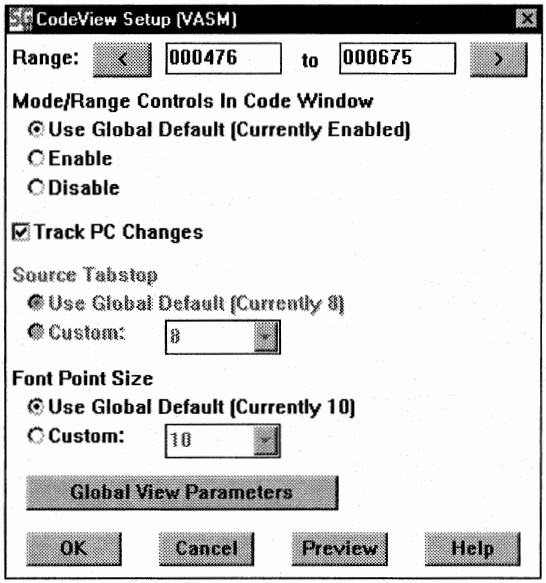
CodeView Setup Window

The CodeView Setup window allows the user to define CodeView preferences for a specific CodeView window. A sample CodeView Setup window for a CodeView window displaying source code is shown below:



To invoke this window, the user must press the "Setup" button from the CodeView window whose preferences are to be changed. In the top border of the window, the name of the CodeView window whose preferences are being modified is shown. Making changes to the preferences shown in this window will only affect this CodeView window.

A sample CodeView Setup window for a VASM CodeView window displaying assembly language code is shown below:



To invoke this window, the user must press the "Setup" button from the VASM CodeView window whose preferences are to be changed. In the top border of the window, the name of the VASM CodeView window whose preferences are being modified is shown. Making changes to the preferences shown in this window will only affect this VASM CodeView window.

The fields shown in the CodeView Setup window are outlined in the sections that follow:

"Mode:" Field (N/A for VASM CodeView Windows)
The "Mode:" field allows the user to define the default mode of the display. Note that the Cursor Tracking Assembly window shows "Cursor Tracking Assembly" in this field which cannot be changed. For additional information on this field, see ***Changing The Display Mode In CodeView Windows***.

"Range:" Field (VASM CodeView Windows Only)
The "Range:" field allows the user to modify the range of addresses shown in a VASM window using the "<" and ">" buttons to increase the range in either direction by 100 hex addresses each time one of these buttons is pressed. The range can also be set by manually entering the desired range in the text boxes provided. For additional information on this field, see ***Viewing Assembly Code In CodeView Windows***.

"Mode/Range Controls In Code Window" Field
This field determines if the "Mode:" or "Range:" fields are shown in the CodeView window. Any one of three different radio buttons can be selected for this option:

> If "Use Global Default" is selected, the display of the "Mode:" and "Range:" fields will be controlled by the CodeView Global View Parameters Window. The current configuration ("Currently Enabled" or "Currently Disabled") as defined in the CodeView Global View Parameters Window is shown for this option.
>
> If "Enable" is selected, the "Mode:" and "Range:" fields will be displayed.
>
> If "Disable" is selected, the "Mode:" and "Range:" fields will not be displayed.

"Track PC Changes" Field (N/A for Cursor Tracking Assembly Window)
This field, when checked, will force the CodeView window to automatically scroll in order to keep the PC visible in the window at all times (as long as the PC remains in the module).

"Source Tabstop" Field (N/A for Cursor Tracking Assembly or VASM Windows)
This field determines the tab stop to be used for the source code displayed in the window. Any one of two different radio buttons can be selected for this option:

If "Use Global Default" is selected, the tab stop value is controlled by the CodeView Global View Parameters Window. The current tab stop value as defined in the CodeView Global View Parameters Window is shown for this option.

If "Custom" is selected, the user can select from a list of available tab stop values by pressing the scroll box for this field and then clicking on the desired tab stop or the tab stop value can be entered manually in the associated text box. Valid tab stops are 0 to 99.

Once the desired changes have been made, the user can preview the changes by pressing the "Preview" button. The "OK" button will save the changes and then close the window and the "Cancel" button will cancel the changes and then close the window. For additional information on the buttons available in this window, see ***CodeView Setup Window Buttons***.

"Font Point Size" Field
This field determines the font size of the source or assembly information that is being displayed in the window. Any one of two different radio buttons can be selected for this option:

If "Use Global Default" is selected, the font size is controlled by the CodeView Global View Parameters Window. The current font size as defined in the CodeView Global View Parameters Window is shown for this option.

If "Custom" is selected, the user can select from a list of available font sizes by pressing the scroll box for this field and then clicking on the desired font size or the font size can be entered manually in the associated text box. Valid font sizes are 1 to 99.

Once the desired changes have been made, the user can preview the changes by pressing the "Preview" button. The "OK" button will save the changes and then close the window and the "Cancel" button will cancel the changes and then close the window. For additional information on the buttons available in this window, see ***CodeView Setup Window Buttons***.

CodeView Setup Window Buttons

The following buttons are available in this window:

OK

The **OK** button will accept the changes made to this window and then close the window.

Cancel

The **Cancel** button will cancel all changes made to this window and then close the window.

Preview

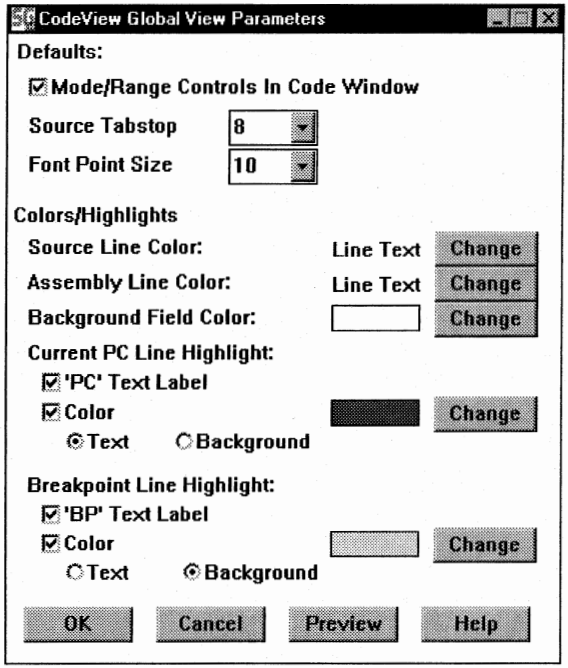
The **Preview** button allows the user to preview the options selected in this window. When this button is pressed, the associated CodeView window will be changed to reflect the selected options.

Help

The **Help** button invokes help for this window.

CodeView Global View Parameters Window

The CodeView Global View Parameters window allows the user to define global CodeView preferences for all CodeView windows. A sample CodeView Global View Parameters window is shown below:



These global preferences will be used until they are changed within the CodeView Setup Window for a particular CodeView window. To invoke this window, the user must press the "Global View Parameters" button from the CodeView Setup Window or select the "Config/Preferences/CodeView View..." menu item.

The fields shown in the CodeView Global View Parameters window are outlined in the sections that follow:

"Defaults:" Field

The "Defaults:" field determines default values for all CodeView windows. An overview of these defaults follows:

If the "Mode/Range Controls In Code Window" field is checked, then the user will have the ability to change the "Mode" and "Range" options from within the CodeView windows.

The "Source Tabstop" field determines the tab stop to be used for the source code displayed in the window.

The "Font Point Size" field determines the font size of the source or assembly information that is being displayed in the window.

"Colors/Highlights:" Field

The "Colors/Highlights:" field determines what colors are shown in CodeView windows. An overview of these defaults follows:

The "Source Line Color:" field allows the user to change the color of the source lines displayed in CodeView windows by pressing the "Change" button next to this field. For additional information on how to change this color, see ***CodeView Color Selection Window***.

The "Assembly Line Color:" field allows the user to change the color of the assembly language lines displayed in CodeView windows by pressing the "Change" button next to this field. For additional information on how to change this color, see ***CodeView Color Selection Window***.

The "Background Field Color:" field allows the user to change the color of the CodeView window background by pressing the "Change" button next to this field. For additional information on how to change this color, see ***CodeView Color Selection Window***.

The "Current PC Line Highlight:" fields allow the user to change how the line pointed to by the program counter (PC) is displayed. The first check box is labeled "PC Text Label" and this box, when checked, will designate the line pointed to by the current PC to be shown with a "PC" label to the left of the line. The second check box is labeled "Color" and it allows the user to select a color for this line. The two radio buttons determine if the "Text" or the "Background" of the PC line is the selected color. If "Background" is selected, the color of the text itself will be defined by the "Source Line Color:" field. Note that both of these boxes must be checked for the user to have total control of the appearance of the PC line. If the "PC Text Label" box is not checked, the current color mode for the line will no longer be changeable and the PC line color will stay at it's present value. If the "Color" field is not checked, the PC line will be designated with "PC" to the left of the line and the color will be defined by the "Source Line Color:" field.

The "Breakpoint Line Highlight:" fields allow the user to change how the lines that have been set to be breakpoints are displayed. The first check box is labeled "BP Text Label" and this box, when checked, will designate the lines set as breakpoints to be shown with a "BP" label to the left of the lines. The second check box is labeled "Color" and it allows the user to select a color for these lines. The two radio buttons determine if the "Text" or the "Background" of the BP lines are the selected color. If "Background" is selected, the color of the text itself will be defined by the "Source Line Color:" field. Note that both of these boxes must be checked for the user to have total control of the appearance of the BP lines. If the "BP Text Label" box is not checked, the current color mode for the line will no longer be changeable and the BP line color will stay at it's present value. If the "Color" field is not checked, the BP line will be designated with "BP" to the left of the line and the color will be defined by the "Source Line Color:" field.

Once the desired changes have been made, the user can preview the changes by pressing the "Preview" button. The "OK" button will save the changes and then close the window and the "Cancel" button will cancel the changes and then close the window. For additional information on the buttons available in this window, see ***CodeView Global View Parameters Window Buttons***.

CodeView Global View Parameters Window Buttons

The following buttons are available in this window:

OK

The **OK** button will accept the changes made to this window and then close the window.

Cancel

The **Cancel** button will cancel all changes made to this window and then close the window.

Preview

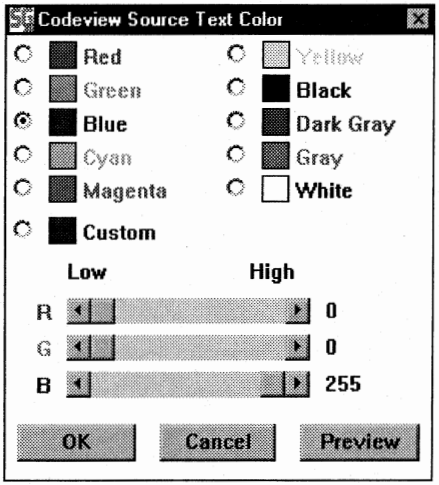
The **Preview** button allows the user to preview the options selected in this window. When this button is pressed, the associated CodeView window will be changed to reflect the selected options.

Help

The **Help** button invokes help for this window.

CodeView Color Selection Window

When the "Change" button is pressed from within one of the CodeView preference windows, a CodeView Color Selection window is opened. A sample CodeView Color Selection window is shown below:



The name of this newly opened window will reflect the CodeView window field whose color is being changed. For example, if the "Change" button for the "Source Line Color:" field is pressed, a color selection window named "CodeView Source Text Color" will be opened.

From this window, the color of the selected field can be selected by clicking on one of the pre-defined colors or by defining a custom color. To define a custom color, click on the "Custom" radio button and then use the color slide bars for red (R), green (G), and blue (B) to define the custom color. The user can click in the bar field to increase or decrease the value by 20 on each click. The slide bar can also be moved by clicking on it, holding the mouse down, and then sliding it to the desired value for that color which is automatically updated as the slide bar is moved. Note that the custom color will change dynamically in the color box located to the left of the "Custom" label as the color is changed. This provides a convenient way to preview the custom color being generated.

Once the desired changes have been made, the user can preview the changes by pressing the "Preview" button. The "OK" button will save the changes and then close the window and the "Cancel" button will cancel the changes and then close the window. For additional information on the buttons available in this window, see ***CodeView Color Selection Window Buttons***.

CodeView Color Selection Window Buttons

The following buttons are available in this window:

OK

The **OK** button will accept the changes made to this window and then close the window.

Cancel

The **Cancel** button will cancel all changes made to this window and then close the window.

Preview

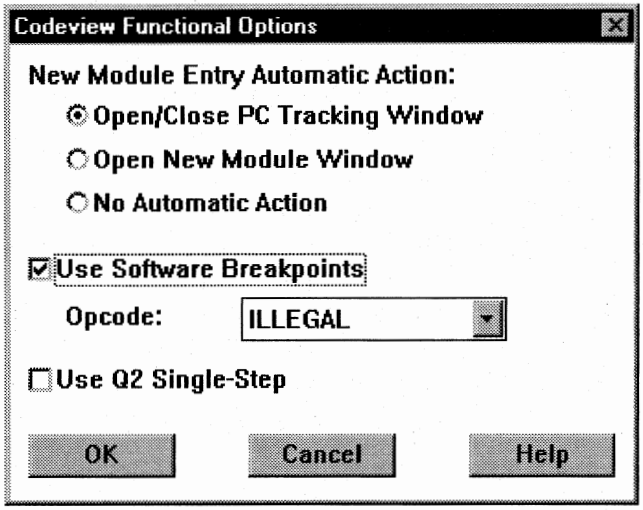
The **Preview** button allows the user to preview the options selected in this window. When this button is pressed, the associated CodeView window will be changed to reflect the selected options.

Color Slide Bar

The **Color Slide Bar** is used to define a custom color. The user can click in the bar field to increase or decrease the value by 20 on each click. The slide bar can also be moved by clicking on it, holding the mouse down, and then sliding it to the desired value for that color which is automatically updated as the slide bar is moved.

CodeView Functional Options Window

The CodeView Functional Options window allows the user to determine how the CodeView window will function in certain instances. A sample CodeView Functional Options window is shown below:



This window is invoked by selecting the "Config/Preferences/CodeView Function..." menu item. The fields shown in the CodeView Functional Options window are outlined in the sections that follow:

"New Module Entry Automatic Action:" Field

The "New Module Entry Automatic Action:" field determines default automatic actions for all CodeView windows by selecting one of three radio buttons. An overview of these default automatic options follows:

> If the "Open/Close PC Tracking Window" field is selected, when a jump to another module or assembly language area occurs (when single-stepping, for example), a PC Tracking window will automatically be opened and will track the area around the PC until the PC returns to the original CodeView window. At that time, the opened PC Tracking window will close automatically. This opening/closing operation is completely automatic and requires no input from the user. For additional information on the operation of the PC Tracking window, see ***Using The PC Tracking Window***.

If the "Open New Module Window" field is selected, when a jump to another module or assembly language area occurs (when single-stepping, for example), a new CodeView or VASM window will be opened for that module or assembly language area. However, these newly opened windows will not automatically close when the PC leaves the window (as was the case in the option described above). Note that this option opens a CodeView window specific to the code being accessed and this window is NOT a PC Tracking window.

If the "No Automatic Action" field is selected, when a jump to another module or assembly language area occurs (when single-stepping, for example), no new CodeView windows will be opened and focus will remain in the current CodeView window.

"Use Software Breakpoints" Field (SPS-2000 Series and Certain HMI-200 Series Emulators Only)

The "Use Software Breakpoints" field determines if software breakpoints are to be used instead of the available hardware based breakpoints. If this box is checked, the user has the option of defining the software breakpoint instruction that is actually written to memory. The default software breakpoint instruction to be used is shown in the "Opcode:" field. The scroll box button for this field may enable the user to select from other instructions that can be used for setting software breakpoints. If multiple instructions are shown, simply click on the one to be used for setting software breakpoints.

"Use Q2 Single-Step" Field (N/A For BMDs or Simulators)

The "Use Q2 Single-Step" field uses the in-circuit emulator's "SS -Q2" command to perform single-stepping operations. This normally results in faster single-step operations due to no other windows (Register, Watch, etc.) being updated between steps. These other windows will be updated with their current values when single-stepping operations are terminated.

For additional information on the buttons available in this window, see ***CodeView Functional Options Window Buttons***.

CodeView Functional Options Window Buttons

The following buttons are available in this window:

OK

The **OK** button will accept the changes made to this window and then close the window.

Cancel

The **Cancel** button will cancel all changes made to this window and then close the window.

Help

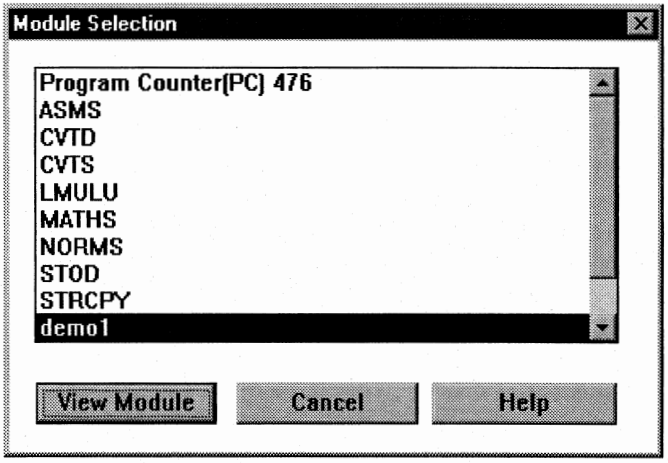
The **Help** button invokes help for this window.

Loading Source Code Into CodeView Windows

This section will describe the steps required to load source code into a CodeView window.

Module Selection Window

The first step in loading source code information into a CodeView window is to open the Module Selection Window by selecting the "CodeView/Modules..." menu item, pressing the "Ctrl+O" key combination, or from within an open CodeView window by pressing the "Module" button. An example Module Selection window is shown below:



The Module Selection window allows the user to choose a module that will be loaded into a CodeView window. Once a module is chosen in this manner, CodeView window operations can be performed on it. An assembly window showing the information at the current program counter location can also be loaded.

To choose a module, place the mouse pointer on the desired module and then press the left mouse button. The highlight bar will move to that module. To then load this module into a CodeView window, press the "View Module" button. A module can also be loaded by double-clicking on the module.

Module Selection Window Buttons

The following buttons are available in the Module Selection window:

View Module

The View Module button will load the highlighted module into a CodeView window.

Cancel

The **Cancel** button will cancel the module selection process and will close the Module Selection window.

Help

The **Help** button will provide help for this window.

Viewing Multiple CodeView Windows

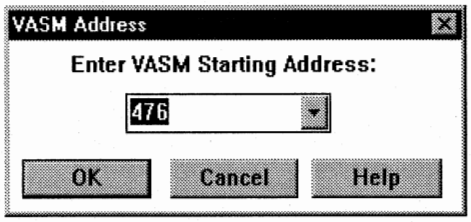
Multiple CodeView windows can be opened at any time, including multiple copies of the same module. The mechanism for opening multiple CodeView windows is no different than for opening a single CodeView window as was discussed in the preceding sections. The number of separate CodeView windows that can be opened at any one time is limited only by the memory in your computer.

Having the ability to view multiple CodeView windows of the same module is handy if you wish to view two different areas of the code at the same time. The top border of the CodeView window will show the name of the module being viewed. If multiple CodeView windows are opened for a particular module, the duplicate windows are designated in the top window border with the label "module:n" where "module" is the module that is being viewed and "n" is the nth window that has been opened for that module. For example, "demo1:4" would designate that window as being the 4th CodeView window opened for the module "demo1".

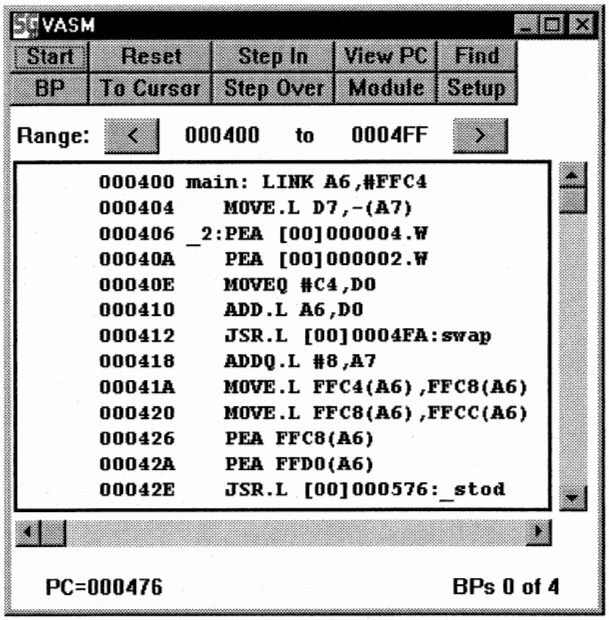
Keep in mind that you have the ability to set breakpoints across all open CodeView windows. For additional information, see ***Setting Breakpoints In CodeView Windows***.

Viewing Assembly Code In CodeView Windows

There are several ways that assembly information can be loaded into a CodeView window. A VASM (View Assembly) window can be shown by selecting the "CodeView/VASM" menu item or by pressing the "Ctrl+V" key combination. The user will then be prompted for a starting address for the VASM window:



A VASM window is a CodeView window that shows information strictly at the assembly language level. The memory address of the instruction, the actual assembly language instruction, and any symbol associated with the address are shown for each line in a VASM window. A sample VASM window is shown below:



The VASM window will initially show 100 hex address locations beginning at the location specified in the VASM Address Window. The user can scroll anywhere in this 100 hex range of addresses. To expand the range of addresses that are shown in the window, the user can use the "<" and ">" buttons to increase the range in either direction by 100 hex addresses each time one of these buttons is pressed. The range can also be set by manually entering the desired range in the boxes provided in the VASM CodeView Setup window. For additional information on how to do this, see ***CodeView Setup Window***.

Perhaps the easiest method of viewing assembly information in a CodeView window is accomplished by simply clicking once on the "Mode:" field and then selecting either "Source With Assembly" or "Assembly Only".

For additional information on how to use the "Mode:" field, see ***Changing The Display Mode In CodeView Windows***.

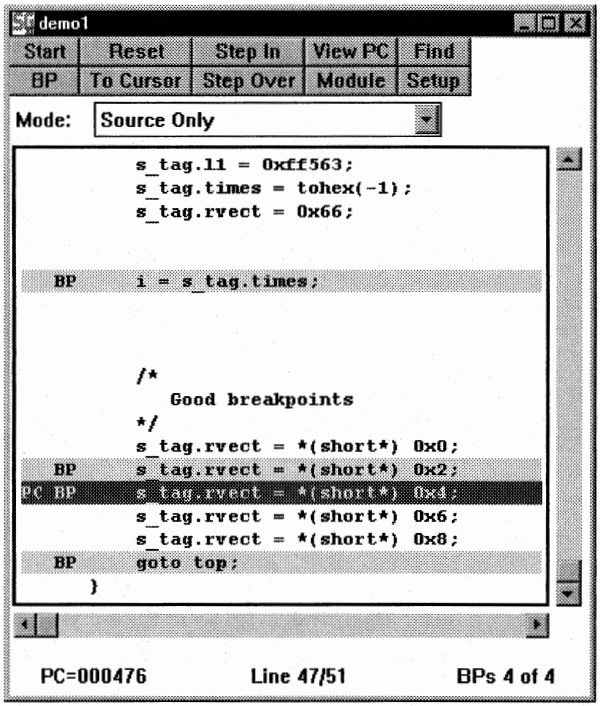
Assembly language can also be loaded into CodeView windows during single-stepping when an assembly language routine is stepped into. For additional information how this process works, see ***Single-Stepping In CodeView Windows***.

Finally, a Cursor Tracking Assembly window can be shown. This window, when active, will show the associated assembly language for the line of source code pointed to by the CodeView window's cursor. For additional information on the operation of this window, see ***Using The Cursor Tracking Assembly Window***.

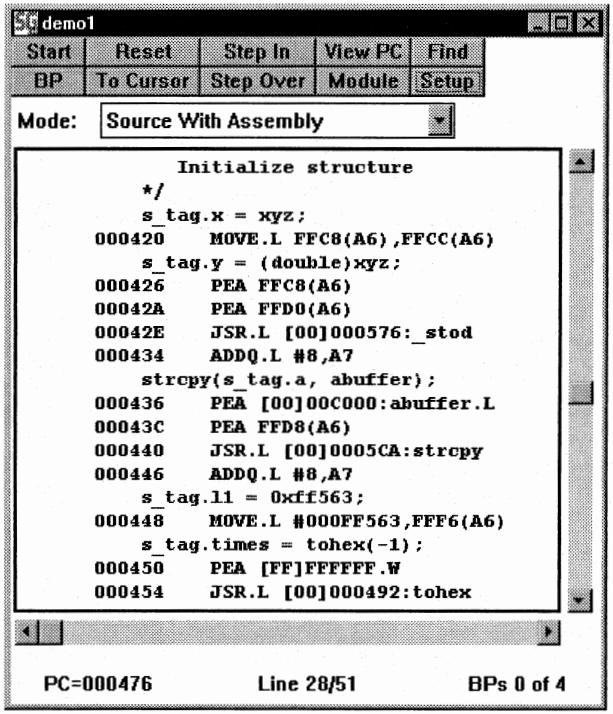
Changing The Display Mode In CodeView Windows

Three different display modes can be set for CodeView windows that are displaying source code: "Source Only", "Source With Assembly", and "Assembly Only". The display of assembly language information in a CodeView window is, of course, always in an "Assembly Only" mode by default.

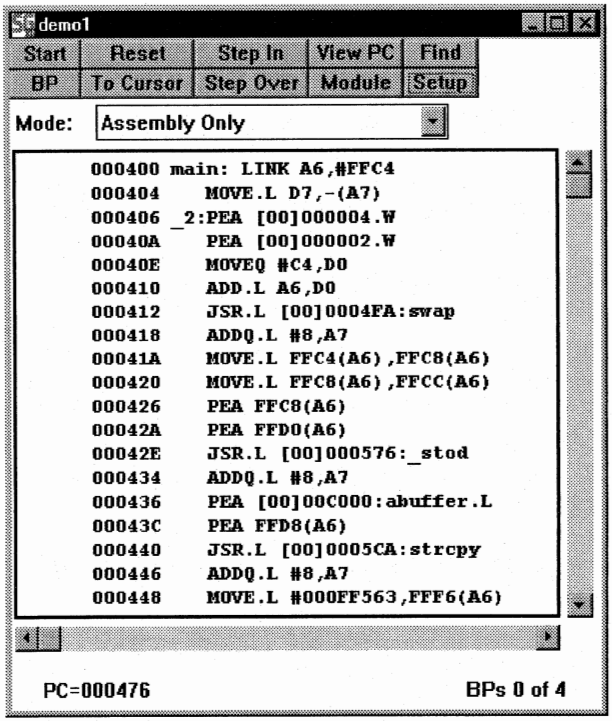
The "Source Only" mode will only display source code information. All assembly language information is suppressed while in this mode. An example CodeView window with the mode set to "Source Only" is shown below:



The "Source With Assembly" mode will show the source code along with the assembly language generated by the source line directly beneath each source line. An example CodeView window with the mode set to "Source With Assembly" is shown below:



The "Assembly Only" mode will show assembly language information only. All source code information is suppressed while in this mode. An example CodeView window with the mode set to "Assembly Only" is shown below:



To change to one of the other modes of operation, simply click on the "Mode:" field and then click on the desired display mode. The CodeView window will then be updated in the newly selected mode.

The display mode can also be defined by default when the CodeView window is first created. For additional information, see ***Setting CodeView Window Preferences***.

Starting Emulation From CodeView Windows

Emulation can be started from a CodeView window in several ways. The first method starts emulation by pressing the "Start" button or by pressing the F5 key. The second method involves highlighting a line in the CodeView window and then pressing the "To Cursor" button. In both cases, emulation will begin from the current program counter location.

Note: When using the CodeView window, emulation must be started by pressing the "Start" button, F5 key, or the "CodeView/Start" menu item. Starting emulation via other methods ("G" command, "Run/Go" menu item, "Go" button) will NOT activate the CodeView window breakpoints and other CodeView specific functions.

Setting Breakpoints In CodeView Windows

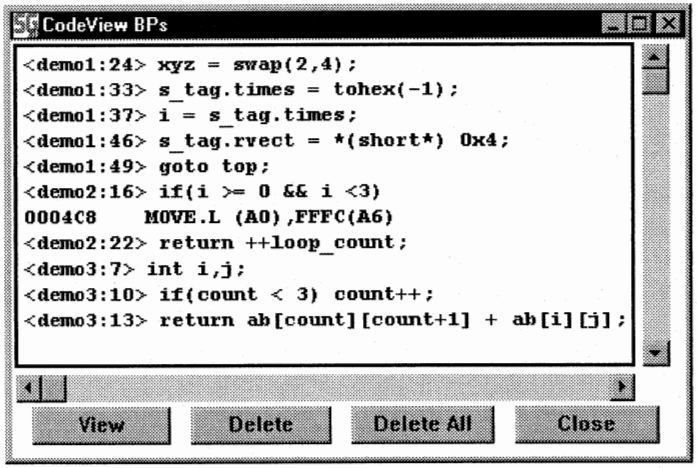
There are several ways that breakpoints can be set in CodeView windows. In the first method, simply click once on the line where you wish the breakpoint to occur. This line will become highlighted. Then, press the "BP" button. This button will toggle the breakpoint setting for the highlighted line. When emulation is started using the "Start" button or F5 key, emulation will begin from the current program counter location and the first occurrence of one of these breakpoints will cause emulation to stop. All of these breakpoints are essentially logically "OR"ed together.

The second method of setting breakpoints is to highlight the line the user wishes to break on and then press the "To Cursor" button. Emulation will begin from the current program counter location and emulation will stop when the highlighted line is reached.

A listing of all breakpoints that have been set with the "BP" button can be obtained from the CodeView BPs Window. If the user attempts to set more than 128 breakpoints, the message "Maximum Number Of Breakpoints [128] Already Set" will be shown.

CodeView BPs Window

The CodeView BPs window is invoked by choosing the "CodeView/List BPs" menu item or by pressing the "Ctrl+L" key combination. This window will show the current breakpoints that have been set from the CodeView windows. An example CodeView BPs Window is shown below:



Up to 128 set breakpoints can be shown in this window. If the user attempts to set a 129th breakpoint, a dialog displaying the message "Maximum Number or Breakpoints [128] Already Set" will be displayed. At this point, no new breakpoints can be set until a currently active breakpoint is removed by highlighting the breakpoint and pressing the "Delete" button. To delete all set breakpoints, the "Delete All" button should be pressed.

To view the module where the set breakpoint resides, the breakpoint of interest should be highlighted and the "View" button should be pressed. This action will open a source or assembly CodeView window that contains the breakpoint.

Breakpoints that are set on source lines are shown in the format:

<module:line#> source_line

Breakpoints that are set on assembly lines are shown in the format:

hex_address assembly_instr

where:

module	is the module that the breakpoint resides in.
line#	is the line number of the module where the breakpoint is set.
hex_address	is the hexadecimal address of the assembly line where the breakpoint is set.
source_line	is the actual source listing of the breakpoint line.
assembly_instr	is the actual assembly instruction listing of the breakpoint line.

CodeView BPs Window Buttons

The following buttons are available in the CodeView BPs window:

View

The **View** button is used to open a CodeView window that contains the highlighted breakpoint line.

Delete

The **Delete** button is used to delete a breakpoint. To do this, highlight the breakpoint to be deleted and then press the **Delete** button.

Delete All

The **Delete All** button removes all breakpoints from the CodeView BPs window. After this button is pressed, no breakpoints will be active.

Close

The **Close** button is used to close the CodeView BPs window without making any changes.

Single-Stepping In CodeView Windows

There are several methods of single-stepping available in CodeView windows. The "Step In" and "Step Over" buttons will step one instruction at the assembly level or one source line at the source level each time they are pressed. Note that stepping can be invoked from any open CodeView window.

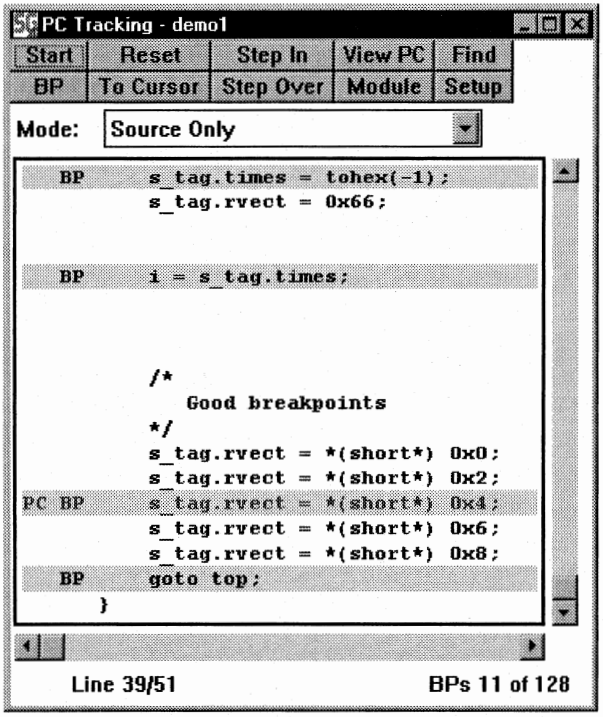
If the user wishes to step into another module when a call to that module is encountered, the "Step In" button should be pressed. When the other module is "stepped into", a new CodeView window will be opened for that module. The same holds true when stepping at the assembly language level in a VASM CodeView window. If a subroutine call is "stepped into", a new VASM window will be opened for the subroutine. The automatic opening and closing of these new CodeView windows during stepping operations can be enabled or disabled by the user via the CodeView preferences. For additional information on how to do this, see ***CodeView Functional Options Window***.

If the user wishes to step over another module when a call to that module is encountered, the "Step Over" button should be pressed. Even though the encountered module will be "stepped over", the code contained in the module call will be executed in real-time. The cursor will then be returned to the next instruction or source line that follows the call instruction. The same holds true when stepping at the assembly language level in a VASM CodeView window.

Note that many windows, including the Register and Watch windows, are updated after each step. This allows the user to track any changes that occur during the stepping process.

Using The PC Tracking Window

The PC Tracking window will automatically load the module or assembly code that contains the current program counter (PC) value. An example PC Tracking Window is shown below:

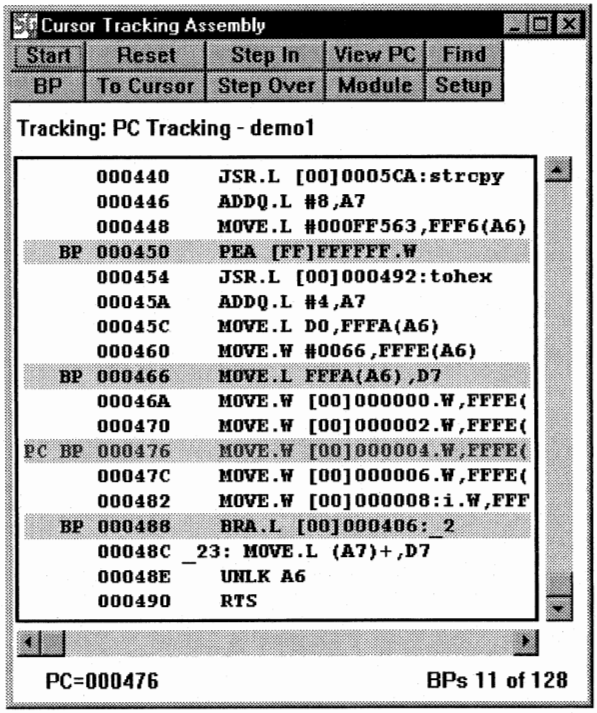


If the PC is located in a source code module, the PC Tracking window will be a typical CodeView window, allowing the user the ability to change display modes (see ***Changing The Display Mode In CodeView Windows*** for additional information on how to change display modes). If the PC is located in an assembly language routine that does not have any associated source code, the PC Tracking window will be a VASM window.

The top border of the PC Tracking window will show the module where the PC currently resides. If the PC resides in an assembly language routine, the top border will say "VASM".

Using The Cursor Tracking Assembly Window

The Cursor Tracking Assembly window will track the CodeView window cursor and will show the assembly language that is associated with the current active CodeView window. An example PC Tracking Window is shown below:



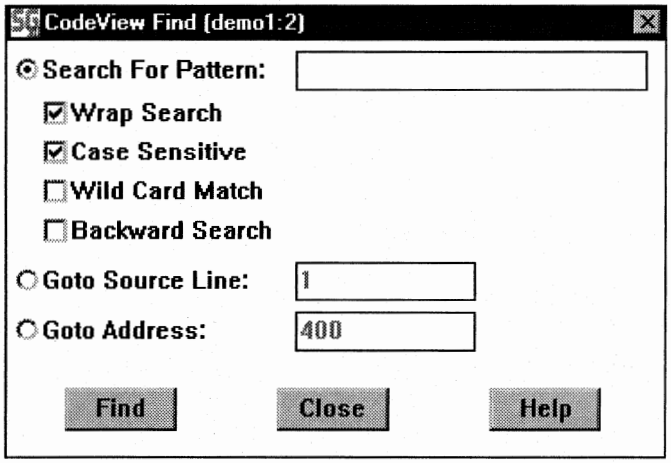
When a source line is highlighted in a CodeView window, the assembly language instructions that are generated from that line of source code are highlighted in the Cursor Tracking Assembly window.

If a VASM window is currently selected as the active CodeView window, then the Cursor Tracking Assembly window will appear to be identical to this VASM window.

The CodeView window being tracked will be shown next to the "Tracking:" label.

Searching The CodeView Window

To search for strings in a CodeView window, press the "Find" button. This will open the CodeView Find window. The CodeView Find window allows the user to search for specific information in a CodeView window. An example CodeView Find window is shown below:



A CodeView Find window is opened by pressing the "Find" button in the CodeView window where the search is to be made. Note that a separate CodeView Find window is available for each open CodeView window. The top border of the CodeView Find window is of the format "CodeView Find (module)" where (module) is the name of the CodeView window from where the Find was initiated.

There are several available search operations that can be initiated from this window. If the CodeView window contains source information, the user can search for a pattern, source line number, or specific memory address. If the CodeView window only contains assembly language information, the user can search for a pattern or specific memory address. The source line option is not available for an assembly only window since the CodeView window contains no source lines. These search operations are mutually exclusive and only one type of search operation can be performed at any one time. All searches begin from the currently highlighted CodeView line.

The "Search For Pattern:" option allows the user to search for a certain pattern in the CodeView window. Any pattern of characters can be searched for simply by entering the desired search string in the provided text box. If the "Wrap Search" box is checked, the search will wrap to the beginning of the CodeView window after the end of the window is reached. If the "Case Sensitive" box is checked, only matches that correspond to the exact text typed in the search box (including capital and lower case characters) will be located during the search. If the "Backward Search" box is checked, the search will proceed from the currently highlighted CodeView line backwards toward the beginning of the CodeView window.

Wild card characters can also be included in the search string by checking the "Wild Card Search" box. The available wild card characters are:

*	Match any (zero or more) characters.
?	Match any single character.
[...]	Match any single character in the enclosed list of characters or ranges. A range is two characters separated by a minus-sign (-), and includes all the characters in between in the ASCII collating sequence.
{str,str,...}	Match any of the Wild Card Strings in the comma-separated list.

The "Goto Source Line:" option allows the user to move the highlight bar to the source line specified in the supplied field. For example, if this option is selected from a CodeView window that is displaying module "Demo 1" and "13" is then entered in the line number field, the highlight bar will move to line 13 in that window and the bottom of that window will show "Line 13/xx" where "xx" is the total number of source lines in the module.

The "Goto Address:" option allows the user to move the highlight bar to the assembly language line corresponding to the address specified in the supplied field. For example, if this option is selected from a CodeView window that is displaying assembly language information and "540" is then entered in the address field, the highlight bar will move to the line of assembly that is located at address "540". Note that the address field is always defined in hexadecimal.

For information on the functions of the buttons in the CodeView Find window, see ***CodeView Find Window Buttons***.

CodeView Find Window Buttons

The following buttons are available in the CodeView Find window:

Find
The **Find** button is used to initiate the search operation.

Close
The **Close** button aborts the search operation and closes the CodeView Find window.

Help
The **Help** button invokes help for this window.

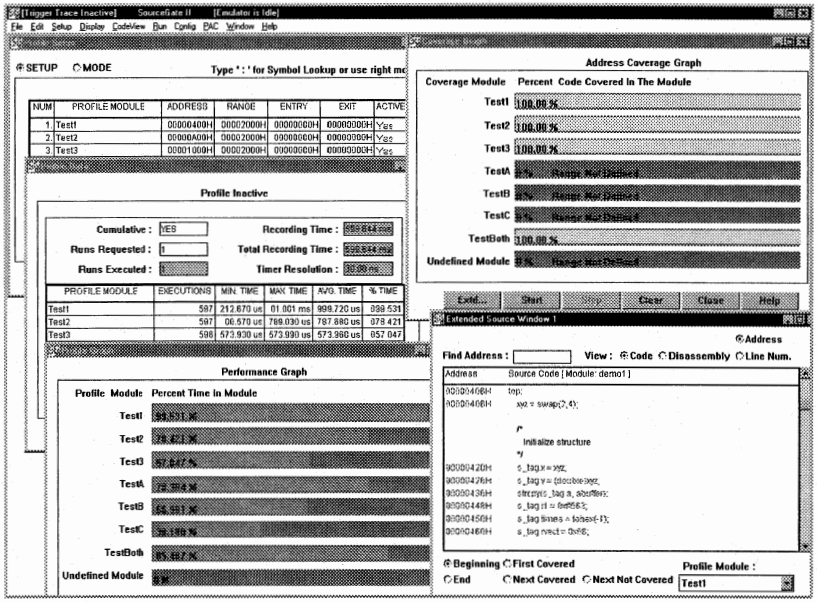
6 Performance Analysis (PAC)

What Is Performance Analysis?

Performance Analysis is a method of measuring code performance. Where does your code spend most of its time? Are all instructions being executed or are some simply "dead code"? With PAC, you can perform runs and get the number of executions of a module, percentage of time spent in a module, minimum/maximum/average time spent in a module, percentage of code covered, and actual addresses that were covered in a module. This information is presented to you in a numerical or histogram format and the numerical information can be saved to a file or printed for future reference.

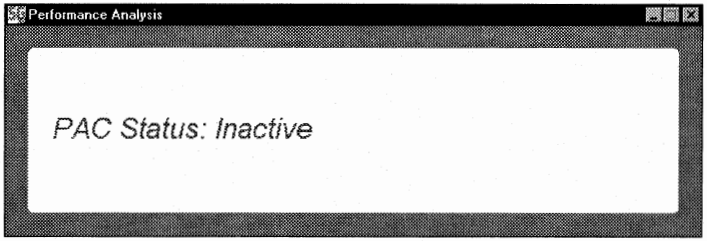
PAC is standard on all SPS-2000 Series emulators and certain HMI-200 Series emulators. It is an option on those HMI-200 Series emulators where it is not standard equipment. Unlike performance analyzers that rely on a statistical software analysis of data, PAC is a hardware card that is located inside of the emulator which provides you with the speed and accuracy of a hardware based system. **Note: PAC is simulated on all HMI Simulators and is not available for Background Mode Debuggers.**

Some results of a sample PAC session are shown below. Each of the features of the PAC system will be shown in the sections that follow.



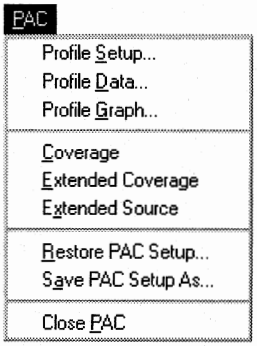
Opening PAC Windows

To enable the PAC system, choose "Display/Performance Analysis" or use the accelerator key combination "Ctrl+N". This will open the main Performance Analysis window which shows the current status of the PAC system:



Note: If the "Display/Performance Analysis" selection is grayed out, your system does not contain PAC.

Once the PAC window is open, an additional menu item labeled "PAC" is shown in the SourceGate II menu bar. Selecting this menu item will show an additional set of selections which allow the user to define what PAC windows are shown:



An overview of each of these menu items is given in the sections that follow.

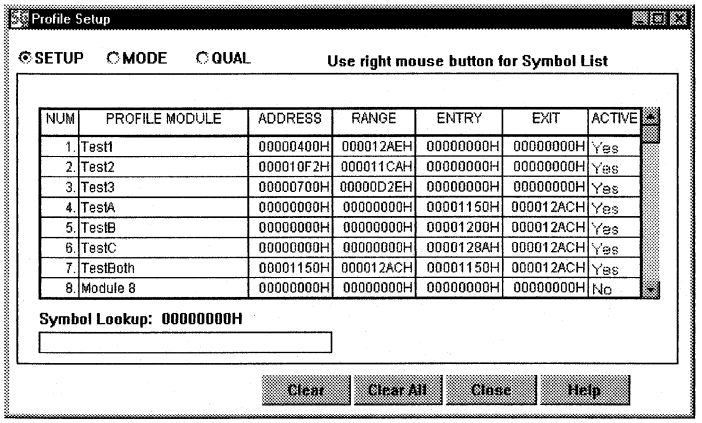
PAC Setup

The first time you use PAC, you will need to set up some parameters so that your analysis runs will capture the data you are interested in. There are three different PAC setup windows, "Setup", "Mode", and "Qualify". These windows are invoked by choosing 'PAC/Profile Setup...'. Each of these windows is discussed in detail in the sections that follow.

Note: SPS-2000 Series emulators do not show a "QUAL" radio button at the top of each PAC Setup window. Therefore, the PAC Setup Qualify window is not available for these systems. Also note that this feature is only supported in certain HMI-200 Series emulators. The "QUAL" selection mechanism will be grayed out in those emulators that do not support this feature.

PAC Profile Setup Window

An example of the Profile Setup window is shown as follows:



This is one of three Profile Setup windows. In this one, selected with the "SETUP" radio button, you define the modules that are to be used in the PAC session. Note that the TAB key (along with the mouse pointer and left mouse button, of course) can be used to move between the different areas and buttons in this window. The arrow keys can be used to move between the address, range, entry, and exit fields. PAC setup information is saved automatically any time changes are made to the PAC Profile Setup window. However, specific setups can be saved to a file and re-loaded at a later date if so desired. See ***Saving and Restoring PAC Setups*** for additional information.

PAC Profile Setup Window Fields

The PAC Profile Setup window has several fields that determine the setup of the PAC system. An overview of these fields is given in the sections that follow.

SETUP Radio Button

The 'SETUP' button, when checked, will show the setups for the modules used in the PAC session.

MODE Radio Button

The "MODE" button, when checked, will show the setups for the record mode, start/stop triggers, record time, and modules selected for new coverage. See ***PAC Profile Mode Setup Window*** for additional information.

QUAL Radio Button

The "QUAL" button, when checked, will show the setups for the read/write and other qualification parameters. This button may not be available on all systems. See ***PAC Setup Qualify Window*** for additional information.

Num Field

The 'Num' field assigns a sequential number to each of the available module definitions. This number always ranges between 1 and 20 and cannot be changed by the user.

Profile Module

The 'Profile Module' field is where the user-defined name of the module is entered. The default names are "Module 1" through "Module 20". You can enter custom names by highlighting the current module name and then typing the desired name.

Address Field

The 'Address' field is where the beginning address for an address-range is entered. The address-range defines a condition whereby, if the processor accesses an address in the defined range, the address will be recorded for processing in the Execution Profile and Coverage Profile windows. Any time the processor accesses a location out of the address-range, the time spent in the module up to that point is calculated and counted as an execution.

The address field must be a smaller number than the corresponding range field or an error condition will occur. The address-range can be any length for timing analysis in the Execution Profile window but must be less than 64K for code coverage analysis in the Coverage Profile window.

Any hexadecimal digit (0-F) is an allowable character for the address field. New values can be entered by using the left mouse button to highlight the fields to be changed and then typing the desired values. In addition, if you wish to enter a symbol, the colon key (:) should be pressed. You can then enter the symbol name in the 'Symbol Lookup' field and SourceGate will load the value associated with that symbol into the field where the insert cursor is currently located. You can also press the right mouse button while in this field. This will invoke the Symbol Selection window.

Range Field

The 'Range' field is where the ending address for an address-range is entered. The address-range defines a condition whereby, if the processor accesses an address in the defined range, the address will be recorded for processing in the Execution Profile and Coverage Profile windows. Any time the processor accesses a location out of the address-range, the time spent in the module up to that point is calculated and counted as an execution.

The range field must be a larger number than the corresponding address field or an error condition will occur. The address-range can be any length for timing analysis in the Execution Profile window but must be less than 64K for code coverage analysis in the Coverage Profile window.

Any hexadecimal digit (0-F) is an allowable character for the range field. New values can be entered by using the left mouse button to highlight the fields to be changed and then typing the desired values. In addition, if you wish to enter a symbol, the colon key (:) should be pressed. You can then enter the symbol name in the 'Symbol Lookup' field and SourceGate will load the value associated with that symbol into the field where the insert cursor is currently located. You can also press the right mouse button while in this field. This will invoke the Symbol Selection window.

Entry Field

The 'Entry' field is where the beginning address for an entry-exit condition is entered. The entry-exit setup defines a condition whereby, when the processor accesses the entry point, a time tag is stored in PAC memory for the entry point. When the exit point is reached, another time tag is stored in PAC memory for the exit point. The difference between the two values is the time given for the particular entry-exit condition and counts as one execution.

Note that entry-exit sessions record all time between the entry and exit points regardless of what address the processor is accessing. Therefore, entry-exit sessions provide the most realistic and useful analysis since they record all operations that occur between the entry and exit points (subroutine calls, interrupt handlers, etc.). There is no requirement that the exit value be larger than the entry value. An entry point of 500H and an exit point of 100H is perfectly valid. In this example, information for the module would begin being recorded when address 500H is accessed and would end when address 100H is accessed.

Any hexadecimal digit (0-F) is an allowable character for the entry field. New values can be entered by using the left mouse button to highlight the fields to be changed and then typing the desired values. In addition, if you wish to enter a symbol, the colon key (:) should be pressed. You can then enter the symbol name in the 'Symbol Lookup' field and SourceGate will load the value associated with that symbol into the field where the insert cursor is currently located. You can also press the right mouse button while in this field. This will invoke the Symbol Selection window.

Exit Field

The 'Exit' field is where the ending address for an entry-exit condition is entered. The entry-exit setup defines a condition whereby, when the processor accesses the entry point, a time tag is stored in PAC memory for the entry point. When the exit point is reached, another time tag is stored in PAC memory for the exit point. The difference between the two values is the time given for the particular entry-exit condition and counts as one execution.

Note that entry-exit sessions record all time between the entry and exit points regardless of what address the processor is accessing. Therefore, entry-exit sessions provide the most realistic and useful analysis since they record all operations that occur between the entry and exit points (subroutine calls, interrupt handlers, etc.). There is no requirement that the exit value be larger than the entry value. An entry point of 500H and an exit point of 100H is perfectly valid. In this example, information for the module would begin being recorded when address 500H is accessed and would end when address 100H is accessed.

Any hexadecimal digit (0-F) is an allowable character for the exit field. New values can be entered by using the left mouse button to highlight the fields to be changed and then typing the desired values. In addition, if you wish to enter a symbol, the colon key (:) should be pressed. You can then enter the symbol name in the 'Symbol Lookup' field and SourceGate will load the value associated with that symbol into the field where the insert cursor is currently located. You can also press the right mouse button while in this field. This will invoke the Symbol Selection window.

Active Field

The 'Active' field determines if the defined module will have information recorded for it during the next PAC recording session. This is a toggle field that can be changed by moving the mouse pointer to this field and pressing the left mouse button. The selections toggle between "Yes" and "No" with "Yes" meaning the module is activated and "No" meaning the module is not activated. **Note that, even though up to 20 modules can be defined, only 8 can be active at any one time. If you attempt to set more than 8 modules to be active, an error message will be displayed.**

Symbol Lookup Field

If you have the insert cursor located in either the Address, Range, Entry, or Exit fields, pressing the colon (:) key will allow a symbol name to be typed into the 'Symbol Lookup' field and SourceGate will load the value associated with that symbol into the field where the insert cursor is currently located. In addition, pressing the right mouse button while in one of these fields will open the Symbol Selection window, from which symbols can be entered. For additional information on using the Symbol Selection window, see ***Working With Symbols*** in Chapter 2.

Also, if you have placed the insert cursor in one of the four above-mentioned fields, a symbol can be entered without using the colon key by pressing the left mouse button while the mouse pointer is in the 'Symbol Lookup' field. After the desired symbol name is typed, followed by pressing the ENTER key, the value associated with that symbol will be loaded into the field where the insert cursor is currently located.

PAC Profile Setup Window Buttons

A description of the functions of the buttons available in this window follows:

Clear

The **Clear** button clears all address, range, entry, and exit point addresses for the selected module. It will also deactivate the selected module by changing the "Active" field for that module to "NO".

Clear All

The **Clear All** button clears all address, range, entry, and exit point addresses for all modules. It will also deactivate all modules by changing the "Active" field for all modules to "NO".

Close

The **Close** button will close this window.

Help

The **Help** button will invoke help for this window.

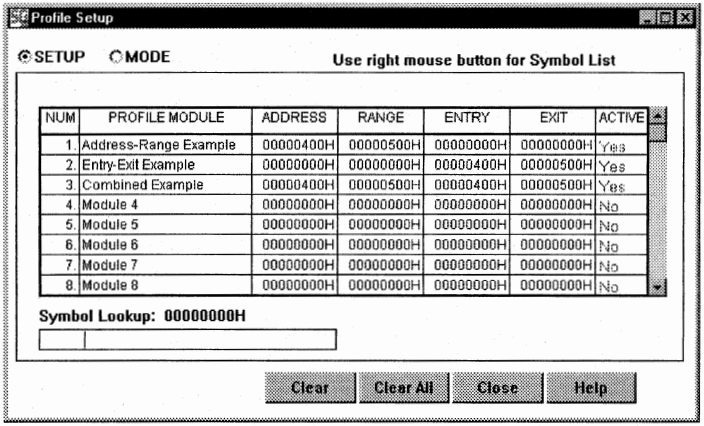
PAC Profile Setup Window Example

An example may help eliminate any confusion about using Address-Ranges, Entry-Exit points, or a combination of the two during a PAC session.

Let's pretend that you have code that you wish to analyze which flows as follows:

Addr.	Instruction	PAC Time Reference
400	NOP	($time_1$)
.	.	
450	JSR 700	($time_2$)
700	NOP	($time_3$)
.	.	
750	RTS	($time_4$)
454	NOP	($time_5$)
.	.	
500	NOP	($time_6$)
502	NOP	($time_7$)

Let's set up three different scenarios using the PAC Profile Setup window and see how PAC performs the analysis for each. The first module will be defined to be an Address-Range between 400H-500H. The second module will be defined to be an Entry-Exit point between 400H-500H. The third module will be a combined Address-Range and Entry-Exit point, each beginning and ending at 400H and 500H, respectively. This example PAC Profile Setup window is shown below:



Module #1 Analysis

PAC will evaluate Module #1 as follows:

Addr.	Instruction	PAC Time Reference	Executions Counted	Time Recorded
400	NOP	($time_1$)		
.	.			
450	JSR 700	($time_2$)		
700	NOP	($time_3$)	1	($time_3$ - $time_1$)
.	.			
750	RTS	($time_4$)		
454	NOP	($time_5$)		
.	.			
500	NOP	($time_6$)		
502	NOP	($time_7$)	1	($time_7$ - $time_5$)

Notice that whenever the program left the defined range, the time spent in the module was calculated and an execution was counted.

To recap, one pass through this code produced the following PAC results:

Number of executions recorded: 2
Time recorded for execution #1: ($time_3$ - $time_1$)
Time recorded for execution #2: ($time_7$ - $time_5$)

Module #2 Analysis
PAC will evaluate Module #2 as follows:

Addr.	Instruction	PAC Time Reference	Executions Counted	Time Recorded
400	NOP	($time_1$)		
.	.			
450	JSR 700	($time_2$)		
700	NOP	($time_3$)		
.	.			
750	RTS	($time_4$)		
454	NOP	($time_5$)		
.	.			
500	NOP	($time_6$)		
502	NOP	($time_7$)	1	($time_7$ - $time_1$)

Notice that all time between the entry and exit points was recorded, even the time spent in the subroutine at address 700. Only when the exit point is reached is an execution counted.

To recap, one pass through this code produced the following PAC results:

Number of executions recorded: 1
Time recorded for execution #1: ($time_7$ - $time_1$)

Module #3 Analysis

PAC will evaluate Module #3 as follows:

Addr.	Instruction	PAC Time Reference	Executions Counted	Time Recorded
400	NOP	($time_1$)		
.	.			
450	JSR 700	($time_2$)		
700	NOP	($time_3$)		($time_3$ - $time_1$)
.	.			
750	RTS	($time_4$)		
454	NOP	($time_5$)		
.	.			
500	NOP	($time_6$)		
502	NOP	($time_7$)	1	($time_7$ - $time_5$)

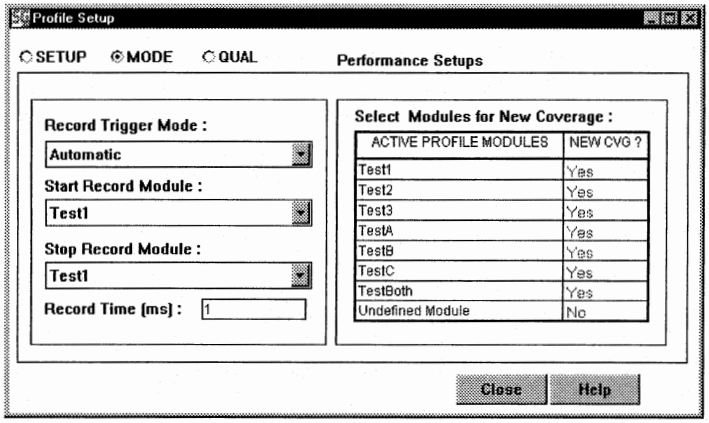
Notice that the time spent outside of the defined range was not recorded. Also note that an execution was not counted until the end of the range was reached.

To recap, one pass through this code produced the following PAC results:

Number of executions recorded: 1
Time recorded for execution #1: ($time_3$ - $time_1$) + ($time_7$ - $time_5$)

PAC Profile Mode Setup Window

If you click on the "MODE" check box, you will see a setup window that defines the record mode, start/stop triggers, record time, and modules selected for new coverage. Note that the TAB key (along with the mouse pointer and left mouse button, of course) can be used to move between the different areas and buttons in this window. An example of this "MODE" window follows:



PAC Profile Mode Setup Window Fields

The PAC Profile Mode Setup window has several fields that determine the mode of operation of the PAC system. An overview of these fields is given in the sections that follow.

SETUP Radio Button

The 'SETUP' button, when checked, will show the setups for the modules used in the PAC session. See ***PAC Profile Setup Window*** for additional information.

MODE Radio Button

The "MODE" button, when checked, will show the setups for the record mode, start/stop triggers, record time, and modules selected for new coverage.

QUAL Radio Button

The "QUAL" button, when checked, will show the setups for the read/write and other qualification parameters. This button may not be available on all systems. See ***PAC Setup Qualify Window*** for additional information.

Record Trigger Mode Field
The 'Record Trigger Mode' field allows you to define how the PAC system will be triggered. The choices are:

Automatic (default):
The analyzer will begin collecting data immediately after the START button is pressed.

Start on Module:
The analyzer will begin collecting data when the module defined in the 'Start Record Module' field is accessed. This option is sometimes needed to keep from using up the limited amount of time tag RAM in the PAC system.

Start/Stop on Module:
The analyzer will begin collecting data when the module defined in the 'Start Record Module' field is accessed and will stop collecting data when the module defined in the 'Stop Record Module' field is accessed. This option is sometimes needed to keep from using up the limited amount of time tag RAM in the PAC system.

Start Record Module Field
If the 'Record Trigger Mode' field contains either "Start on Module" or "Start/Stop on Module" this field determines the module that will start the analyzer collecting data. If the 'Record Trigger Mode' field contains "Automatic", this field is ignored. The default module placed here will be the first enabled module listed in the Profile Setup window.

Stop Record Module Field
If the 'Record Trigger Mode' field contains "Start/Stop on Module" this field determines the module that will stop the analyzer from collecting data. If the 'Record Trigger Mode' field contains "Automatic" or "Start on Module", this field is ignored.

Record Time Field
The 'Record Time' field determines how long the analyzer will be run for each pass. It is defined in milliseconds with a default value of 60000.

Select Modules For New Coverage Field
The 'Select Modules for New Coverage' field lists the active modules (as defined in the Profile Setup window) and allows you to enable new coverage information to be generated for them. This field can be toggled between "YES" and "NO" by moving the mouse pointer to the "NEW CVG" field and pressing the left mouse button. All of the listed modules will toggle between "YES" for establish new coverage on each run to "NO" for do not establish new coverage on each run but instead append to the existing coverage information.

PAC Profile Mode Setup Window Buttons

A description of the functions of the buttons available in this window follows:

Close

The **Close** button will close the PAC Profile Mode Setup window.

Help

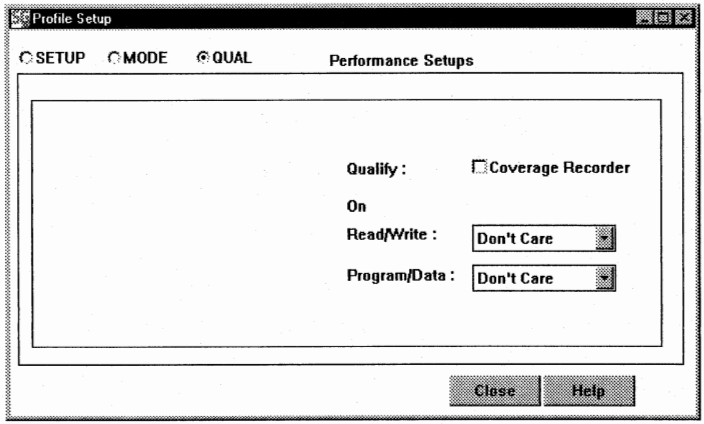
The **Help** button will invoke help for the PAC Profile Mode Setup window.

Scroll Box

The **Scroll Box** button will allow the user to scroll through the available choices for certain fields.

PAC Setup Qualify Window

If you click on the "QUAL" check box, you will see a setup window that defines read/write and other qualification parameters. Note that the TAB key (along with the mouse pointer and left mouse button, of course) can be used to move between the different areas and buttons in this window. An example of this "QUAL" window follows:



Note: SPS-2000 Series emulators do not show a "QUAL" radio button at the top of each PAC Setup window. Therefore, the PAC Setup Qualify window is not available for these systems. Also note that this feature is only supported in certain HMI-200 Series emulators. The "QUAL" selection mechanism will be grayed out in those emulators that do not support this feature.

PAC Setup Qualify Window Fields

The PAC Setup Qualify window has several fields that define read/write and other qualification parameters. An overview of these fields is given in the sections that follow.

SETUP Radio Button

The 'SETUP' button, when checked, will show the setups for the modules used in the PAC session. See ***PAC Profile Setup Window*** for additional information.

MODE Radio Button

The "MODE" button, when checked, will show the setups for the record mode, start/stop triggers, record time, and modules selected for new coverage. See ***PAC Profile Mode Setup Window*** for additional information.

QUAL Radio Button
The "QUAL" button, when checked, will show the setups for the read/write and other qualification parameters. This button may not be available on all systems.

'Qualify' Check Box
There is a 'Qualify' check box available for the Coverage recorder. Checking this box activates qualification. This 'Coverage Recorder' qualify check box will allow the defined qualification to influence the coverage run.

'Read/Write' Field
This field, along with the 'Program/Data' field, determines what information is recorded when using the Coverage Recorder. The choices are "Read", "Write", and "Don't Care". If "Read" is selected, only memory reads are recorded as part of the run. If "Write" is selected, only memory writes are recorded as part of the run. If "Don't Care" is selected, both memory reads and memory writes are recorded as part of the run.

In previous versions of SourceGate, there was no way to differentiate between a opcode/operand fetch and a data fetch. This caused some of the Coverage data to appear incorrect. With the capability to select read/write operations, along with the ability to select the memory space, more "qualified" data can be captured with the performance system.

'Program/Data' Field
This field, along with the 'Read/Write' field, determines what information is recorded when using the Coverage Recorder. The choices are "Program", "Data", and "Don't Care". If "Program" is selected, only program memory accesses are recorded as part of the run. If "Data" is selected, only data memory accesses are recorded as part of the run. If "Don't Care" is selected, both program memory and data memory accesses are recorded as part of the run.

In previous versions of SourceGate, there was no way to differentiate between a opcode/operand fetch and a data fetch. This caused some of the Coverage data to appear incorrect. With the capability to select the memory space, along with the ability to select read/write operations, more "qualified" data can be captured with the performance system.

PAC Setup Qualify Window Buttons

A description of the functions of the buttons available in this window follows:

Close

The **Close** button will close the PAC Setup Qualify window.

Help

The **Help** button will invoke help for the PAC Setup Qualify window.

Scroll Box

The **Scroll Box** button will allow the user to scroll through the available choices for certain fields.

Saving and Restoring PAC Setups

PAC setup information is saved automatically anytime changes are made to the PAC Profile Setup window. However, specific setups can be saved to a file and re-loaded at a later date if so desired. The following sections describe how to do this.

Saving A PAC Setup

The 'PAC/Save PAC Setup As...' command will save all PAC parameters to a file. After invoking this command, you will be prompted for the name of the file to save the PAC parameters to. The default extension of ".SET" should be used. This file can then be recalled using the 'PAC/Restore PAC Setup...' command.

Restoring A Previously Saved PAC Setup

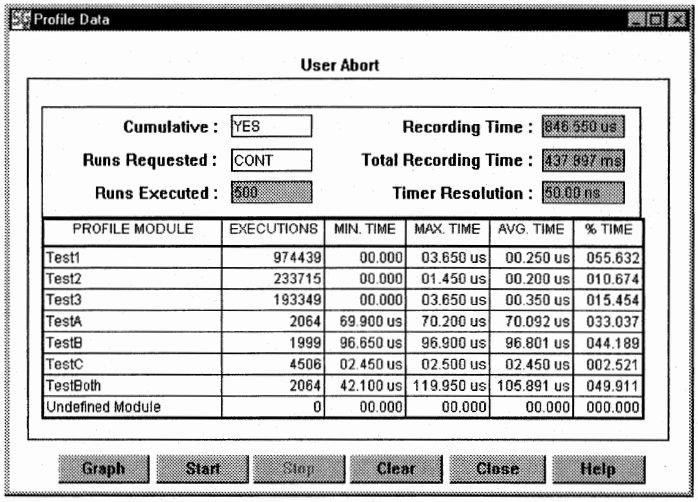
The 'PAC/Restore PAC Setup...' command will restore PAC parameters that had been saved to a file with the ".SET" extension using the 'PAC/Save PAC Setup As..." command. After invoking this command, you will be prompted for the name of the file to be restored. After loading this file, all PAC parameters should reflect the information stored in the file.

Recording PAC Profile Data

Once you have set up the PAC system, you are then ready to perform a run and analyze your data. You can begin a PAC Profile session from either the PAC Execution Profile window, or the PAC Execution Profile Graph window. An overview of each of these windows is given in the sections that follow.

PAC Execution Profile Window

The PAC Execution Profile window allows you to start or stop a PAC session and show the results. The number of executions, minimum time, maximum time, average time, percentage of total record time for each defined module, and other fields are shown here. You can open the PAC Execution Profile window by choosing 'PAC/Profile Data...'. An example PAC Execution Profile window is shown as follows:


Profile Data
User Abort
Cumulative : YES
Runs Requested : CONT
Runs Executed : 500
Recording Time : 846.550 us
Total Recording Time : 437.997 ms
Timer Resolution : 50.00 ns
PROFILE MODULE | EXECUTIONS | MIN. TIME | MAX. TIME | AVG. TIME | % TIME
Test1 | 974439 | 00.000 | 03.650 us | 00.250 us | 055.632
Test2 | 233715 | 00.000 | 01.450 us | 00.200 us | 010.674
Test3 | 193349 | 00.000 | 03.650 us | 00.350 us | 015.454
TestA | 2064 | 69.900 us | 70.200 us | 70.092 us | 033.037
TestB | 1999 | 96.650 us | 96.900 us | 96.801 us | 044.189
TestC | 4506 | 02.450 us | 02.500 us | 02.450 us | 002.521
TestBoth | 2064 | 42.100 us | 119.950 us | 105.891 us | 049.911
Undefined Module | 0 | 00.000 | 00.000 | 00.000 | 000.000
Graph
Start
Stop
Clear
Close
Help

PAC Execution Profile Window Fields

The PAC Execution Profile window has several fields that provide information about the analysis runs that have been performed. An overview of these fields is given in the sections that follow.

PAC Execution Profile 'Cumulative' Field

The 'Cumulative' field allows the user to define if the executions, min/max/avg times, and % times are calculated as a summation of the data from all of the runs "YES" or just a representation of the latest run "NO".

To change this field, highlight the existing entry and then type "YES" or "NO".

PAC Execution Profile 'Runs Requested' Field
The 'Runs Requested' field allows the user to specify the number of runs that are to be performed. A specific number can be entered or "0" can be entered for continuous reruns (PAC will run until the "Stop" button is pressed).

PAC Execution Profile 'Runs Executed' Field
The 'Runs Executed' field tells the user how many runs have been executed. **Note: This is an information only field and cannot be changed by the user.**

PAC Execution Profile 'Recording Time' Field
The 'Recording Time' field tells the user the recording time for the last run. **Note: This is an information only field and cannot be changed by the user.**

PAC Execution Profile 'Total Recording Time' Field
The 'Total Recording Time' field tells the user the total recording time for the entire session. This number will be the same value as the 'Recording Time' field if the 'Cumulative' field is set to "NO" and will be the total recording time of all runs if the 'Cumulative' field is set to "YES". **Note: This is an information only field and cannot be changed by the user.**

PAC Execution Profile 'Timer Resolution' Field
The 'Timer Resolution' field shows the resolution used by the PAC system. This field is set at 50 nanoseconds for HMI-200 Series emulators and HMI Simulators. It is set at 30 nanoseconds for SPS-2000 Series emulators. **Note: This is an information only field and cannot be changed by the user.**

PAC Execution Profile 'Profile Module' Field
The 'Profile Module' field shows the active modules as defined in the Profile Setup window and lists the results of their PAC session including number of executions, min/max/avg time, and percentage of time spent in each module. Modules that have been disabled in the Profile Setup window will not be shown in this window. **Note: This is an information only field and cannot be changed by the user.**

PAC Execution Profile 'Executions' Field
The 'Executions' field shows the number of times that the analyzer processed information in the listed module. If the module has an address-range defined, each time the processor accesses a memory location outside of the defined address-range, a time tag is put into PAC memory for that location and an execution is counted. Similarly, if an entry-exit point has been defined for a module, any time the entry point is reached followed by the exit point, this is counted as one execution.

The number of executions shown is either a total of all of the runs performed or just the last run. Which one is shown depends on the 'Cumulative' field. A "YES" value in the 'Cumulative' field will force a cumulative total of all executions in the 'Executions' field. A "NO" value in the 'Cumulative' field will force the 'Executions' field to contain the number of executions for only the last run. **Note: This is an information only field and cannot be changed by the user.**

PAC Execution Profile 'Min. Time' Field

The 'Min. Time' field shows the smallest time it took for an execution to occur in the specified module. The value shown is the smallest value obtained in all of the runs performed if the 'Cumulative' field is set to "YES" and is the result of the latest run if the 'Cumulative' field is set to "NO". **Note: This is an information only field and cannot be changed by the user.**

PAC Execution Profile 'Max. Time' Field

The 'Max. Time' field shows the largest time it took for an execution to occur in the specified module. The value shown is the largest value obtained in all of the runs performed if the 'Cumulative' field is set to "YES" and is the result of the latest run if the 'Cumulative' field is set to "NO". **Note: This is an information only field and cannot be changed by the user.**

PAC Execution Profile 'Avg. Time' Field

The 'Avg. Time' field shows the average time it took for an execution to occur in the specified module. The value shown is the average value obtained in all of the runs performed if the 'Cumulative' field is set to "YES" and is the result of the latest run if the 'Cumulative' field is set to "NO". **Note: This is an information only field and cannot be changed by the user.**

PAC Execution Profile '% Time' Field

The '% Time' field shows the percentage of the recording time that was spent in the specified module. The value shown is the percentage of the total recording time obtained in all of the runs performed if the 'Cumulative' field is set to "YES" and is the percentage of the recording time of the latest run if the 'Cumulative' field is set to "NO". **Note: This is an information only field and cannot be changed by the user.**

PAC Execution Profile Window Buttons

The buttons in the PAC Execution Profile window allow the user to perform certain actions. An explanation of all available buttons is given as follows:

Graph

The **Graph** button will invoke the Profile Graph window. The Profile Graph window shows the PAC execution data in a histogram representation.

Start

The **Start** button will start a PAC Execution Profile recording session. The PAC system will begin recording data until the **Stop** button is pressed or until the "Runs Requested" field matches the "Runs Executed" field. While in this window, the result data will be displayed in numerical format.

Stop

The **Stop** button will stop a PAC Execution Profile recording session as soon as the current run is completed. The result data will be displayed in numerical format.

Clear

The **Clear** button clears all PAC data obtained from previous runs for all modules.

Close

The **Close** button will close the PAC Execution Profile window.

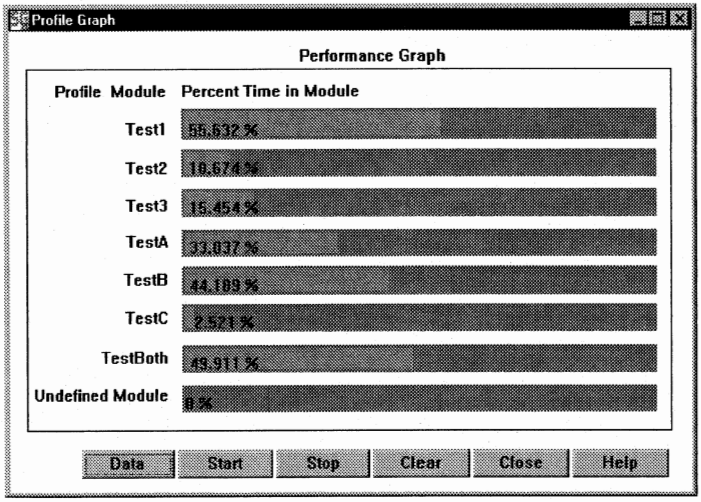
Help

The **Help** button will invoke help for the PAC Execution Profile window.

PAC Execution Profile Graph Window

The PAC Execution Profile Graph window shows the percentage of time spent in each defined module during the PAC run. This percentage is either a percentage of all runs or just the last run depending on whether or not "Cummulative" is selected in the PAC Execution Profile window. If the 'Cummulative' field is set to "YES", the histograms displayed in this window are relative to the total record time for all runs. If the 'Cummulative' field is set to "NO", the histograms displayed are relative to the record time for the latest run. This information is represented in both histogram and numerical percentage form. You can open the PAC Execution Profile Graph window by choosing 'PAC/Profile Graph...'.

An example PAC Execution Profile Graph window is shown as follows:



PAC Execution Profile Graph Window Buttons

A description of the functions of the buttons available in this window follows:

Data

The **Data** button will invoke the Execution Profile window. The Execution Profile window shows the PAC execution data in a numerical representation. **See *PAC Execution Profile Window* for additional information.**

Start
The **Start** button will start a PAC Execution Profile recording session. The PAC system will begin recording data until the **Stop** button is pressed or until the "Runs Requested" field matches the "Runs Executed" field. While in this window, the result data will be displayed in histogram format.

Stop
The **Stop** button will stop a PAC Execution Profile recording session as soon as the current run is completed. The result data will be displayed in histogram format.

Clear
The **Clear** button clears all PAC data obtained from previous runs for all modules.

Close
The **Close** button will close this window.

Help
The **Help** button will invoke help for this window.

Starting A PAC Profile Session

To start a PAC Profile session, press the **Start** button (emulation must be in progress). This will begin the PAC session using the parameters you specified in the Setup windows.

Stopping A PAC Profile Session

To stop a PAC Profile session, press the **Stop** button. This will stop the PAC session and update the PAC Execution Profile with the results from the last run.

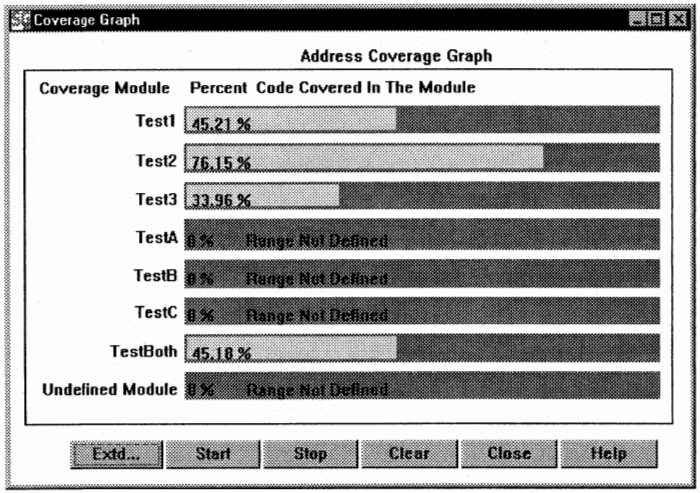
Viewing PAC Coverage Data

Once a PAC run has been completed, you can view the PAC Coverage Profile window which shows the percentage of code that was covered for each module with an address range defined. The PAC Extended Coverage window shows the actual addresses that were covered and not covered for each of these modules. An overview of each of these windows is given in the sections that follow.

PAC Coverage Profile Window

The PAC Coverage Profile window shows the percentage of code that was covered for each module during the PAC run. If the 'NEW CVG' field is set to "YES" in the Profile Mode window, the information shown in this window is the coverage data from the last run. If the 'NEW CVG' field is set to "NO" in the Profile Mode window, the information shown in this window is the coverage data from all prior runs. This information is represented in both histogram and numerical percentage form. You can open the PAC Coverage Profile window by choosing 'PAC/Coverage'.

An example PAC Coverage Profile window is shown as follows:



PAC Coverage Profile Window Buttons

A description of the functions of the buttons available in this window follows:

Extd

The **Extd** button will invoke the Extended Coverage window. This window will show actual covered and not covered addresses and provides a capability to search for specific addresses and print the results to a file.

Start

The **Start** button will start a PAC Coverage Profile recording session. The PAC system will begin recording addresses that are covered and not covered until the **Stop** button is pressed.

Stop

The **Stop** button will stop a PAC Coverage Profile recording session. The PAC Coverage window will then be updated with the percentages of coverage for each module shown in a histogram format.

Clear

The **Clear** button clears all PAC coverage histograms obtained from previous runs for all modules.

Close

The **Close** button will close the PAC Coverage Profile window.

Help

The **Help** button will invoke help for the PAC Coverage Profile window.

Starting A PAC Coverage Profile Session

To start the coverage analysis, press the **Start** button (emulation must be in progress). The PAC system will begin recording addresses that are covered and not covered until the **Stop** button is pressed.

Stopping A PAC Coverage Profile Session

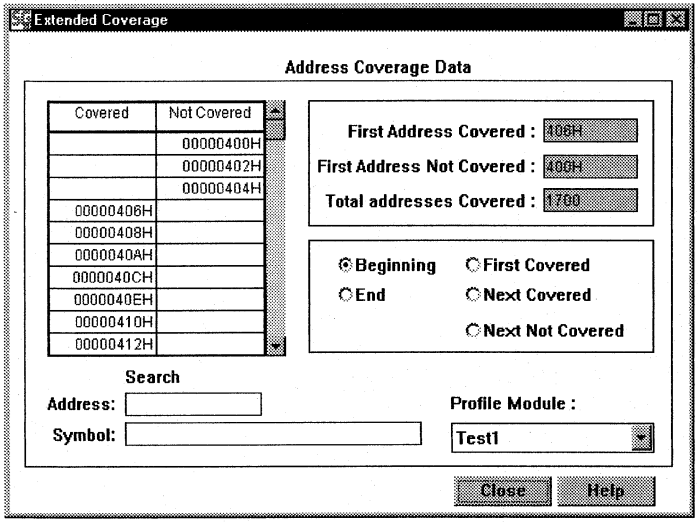
To stop a PAC Coverage Profile session, press the **Stop** button. The PAC Coverage window will then be updated with the percentages of coverage for each module shown in a histogram format.

PAC Extended Coverage Window

The PAC Extended Coverage window shows the actual addresses that were covered and not covered during the coverage run. From this window, you can view all covered and not covered addresses for a specified module along with the first address covered, last address covered, total number of addresses covered, beginning and ending addresses in the module, next covered, next not covered, and search for a specific address or symbol name. You can open the PAC Extended Coverage window by choosing 'PAC/Extended Coverage'. Note that a coverage run must have been performed in order for the 'PAC/Extended Coverage' menu item to be available. If a coverage run has not been performed, this menu item will be grayed out.

If the 'NEW CVG' field is set to "YES" in the Profile Mode window, the information shown in this window is the coverage data from the last run. If the 'NEW CVG' field is set to "NO" in the Profile Mode window, the information shown in this window is the coverage data from all prior runs.

An example PAC Extended Coverage window is shown as follows:



PAC Extended Coverage Window Fields

The PAC Extended Coverage window has several fields that show the results of a PAC extended coverage session. An overview of these fields is given in the sections that follow:

Covered Field

The 'Covered' field shows all addresses that were covered in the previous PAC session. You can use the vertical scroll bar to scroll through the "Covered" data. **Note: This is an information only field and cannot be changed by the user.**

Not Covered Field

The 'Not Covered' field shows all addresses that were covered in the previous PAC session. You can use the vertical scroll bar to scroll through the "Not Covered" data. **Note: This is an information only field and cannot be changed by the user.**

First Address Covered Field

The 'First Address Covered' field lists the first address (in ascending order) that was covered in the last PAC session. **Note: This is an information only field and cannot be changed by the user.**

First Address Not Covered Field

The 'First Address Not Covered' field lists the first address (in ascending order) that was not covered in the last PAC session. **Note: This is an information only field and cannot be changed by the user.**

Total Addresses Covered Field

The 'Total Addresses Covered' field lists the total number of addresses that were covered in the last PAC session. This number is shown in decimal. **Note: This is an information only field and cannot be changed by the user.**

Beginning Radio Button

The 'Beginning' button will take you to the beginning of the extended coverage information. To invoke this function, place the mouse pointer on the 'Beginning' button and then press the left mouse button.

End Radio Button
The 'End' button will take you to the end of the extended coverage information. To invoke this function, place the mouse pointer on the 'End' button and then press the left mouse button.

First Covered Radio Button
The 'First Covered' button will take you to the location of the first covered address in the extended coverage information. To invoke this function, place the mouse pointer on the 'First Covered' button and then press the left mouse button.

Next Covered Radio Button
The 'Next Covered' button will take you to the location of the next covered address in the extended coverage information. To invoke this function, place the mouse pointer on the 'Next Covered' button and then press the left mouse button.

Next Not Covered Radio Button
The 'Next Not Covered' button will take you to the location of the next not covered address in the extended coverage information. To invoke this function, place the mouse pointer on the 'Next Not Covered' button and then press the left mouse button.

Profile Module Field
The 'Profile Module' field allows you to choose which module the extended coverage will be shown for. To choose a module to view, you should press the scroll box button located next to the 'Profile Module' field. You can then choose from all of the modules that are enabled in the Profile Setup window.

Address Search Field
The 'Address Search' field allows you to search for a particular address in the covered/not covered data. To enter this field, move the mouse cursor to this field and then press the left mouse button (or press the TAB key until this field is highlighted). Once inside of the field, an address can be entered in hexadecimal or a symbol name can be entered by typing a colon (:) which will place the cursor in the 'Symbol' field. You can then enter the name of the symbol and SourceGate will enter the corresponding address in the 'Address' field. If the symbol name is invalid, the message "Symbol not found!" will be displayed. You can also press the right mouse button while in this field. This will invoke the Symbol Selection window.

Once the address or symbol name is entered, press the ENTER key to begin the search. If the address value is not found in the extended coverage data, the message "Out of range address" will be displayed.

Symbol Search Field

The 'Symbol Search' field allows you to search for a particular address in the covered/not covered data by entering the corresponding symbol name for the address. To enter this field, move the mouse cursor to this field and then press the left mouse button (or press the TAB key until this field is highlighted). Once inside of the field, you can then enter the name of the symbol and SourceGate will enter the corresponding address in the 'Address' field. If the symbol name is invalid, the message "Symbol not found!" will be displayed.

Once the symbol name is entered, press the ENTER key to begin the search. If the corresponding address value is not found in the extended coverage data, the message "Out of range address" will be displayed.

PAC Extended Coverage Window Buttons

A description of the functions of the buttons available in this window follows:

Close

The **Close** button will close the Extended Coverage window.

Help

The **Help** button will invoke help for the Extended Coverage window.

Scroll Box

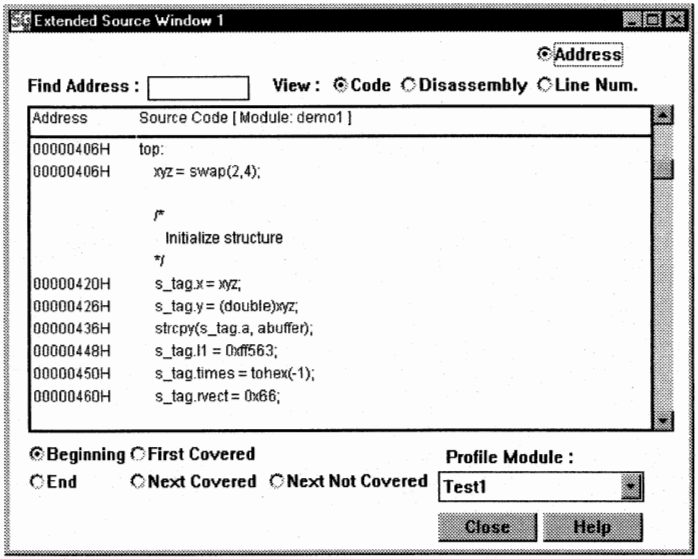
The **Scroll Box** button will allow you to scroll through the available choices for certain fields.

PAC Extended Source Coverage Window

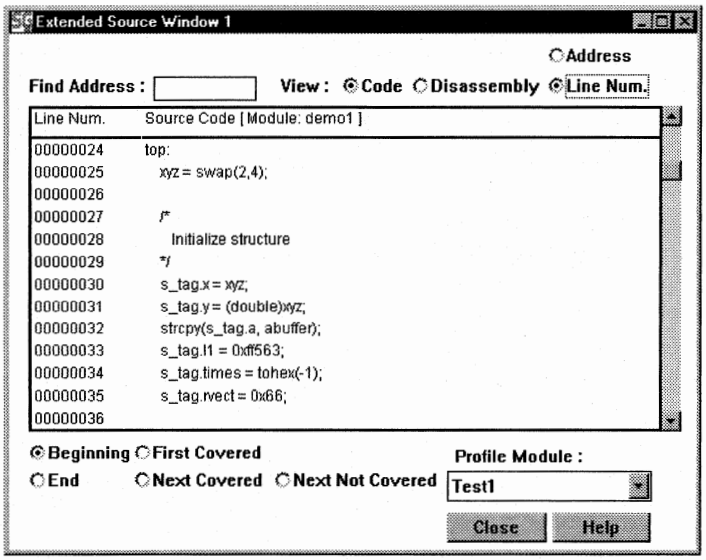
The PAC Extended Source Coverage window shows the actual code that was covered and not covered. From this window, the user can view all covered and not covered code (in either "C" source or disassembly) for a specified module along with beginning and ending code in the module, first covered, next covered, and next not covered. The user can also search for a specific address in the coverage data. Comment lines (or lines that do not produce code) are shown in black, lines that were covered are shown in blue, lines that were not covered are shown in red. When another source file is encountered during the coverage run, "<New File>" is shown. The source code module currently being viewed is shown in the top border of the display window.

If the 'NEW CVG' field is set to "YES" in the Profile Mode window, the information shown in this window is the coverage data from the last run. If the 'NEW CVG' field is set to "NO" in the Profile Mode window, the information shown in this window is the coverage data from all prior runs.

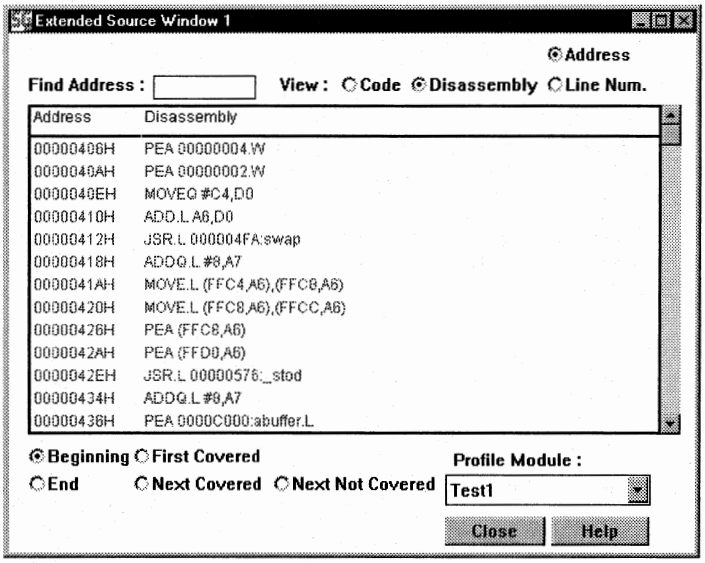
To open this window, choose "PAC/Extended Source". An example PAC Extended Source window displaying addresses and "C" source code coverage is shown as follows:



An example PAC Extended Source window displaying source file line numbers and "C" source code coverage is shown as follows:



An example PAC Extended Source window displaying addresses and assembly language code coverage is shown as follows:



PAC Extended Source Coverage Window Fields

The PAC Extended Source window has several fields that show the results of a PAC extended source coverage session. An overview of these fields is given in the sections that follow:

'View' Radio Buttons

The 'View' options allow the user to specify how the display looks. If 'Code' is chosen, the display will shown "C" source code lines. When 'Code' is being displayed, the user can also choose between 'Address' and 'Line Num'. 'Address' will show the actual address in memory where the machine code for the displayed source line resides. 'Line Num' will show the line number of each line in the source file.

If 'Disassembly' is chosen, the display will show disassembly information. Note that only the 'Address' display option can be used with this mode.

Beginning Radio Button

The 'Beginning' button will take the user to the beginning of the extended source information. To invoke this function, place the mouse pointer on the 'Beginning' button and then press the left mouse button.

End Radio Button

The 'End' button will take the user to the end of the extended source information. To invoke this function, place the mouse pointer on the 'End' button and then press the left mouse button.

First Covered Radio Button

The 'First Covered' button will take the user to the location of the first covered line in the extended source information. To invoke this function, place the mouse pointer on the 'First Covered' button and then press the left mouse button.

Next Covered Radio Button

The 'Next Covered' button will take the user to the location of the next covered line in the extended source information. To invoke this function, place the mouse pointer on the 'Next Covered' button and then press the left mouse button.

Next Not Covered Radio Button

The 'Next Not Covered' button will take the user to the location of the next not covered line in the extended source information. To invoke this function, place the mouse pointer on the 'Next Not Covered' button and then press the left mouse button.

'Profile Module' Field
The 'Profile Module' field allows the user to choose which module the extended source will be shown for.

To choose a module to view, the user should press the **Scroll Box** button located next to the 'Profile Module' field. The user can then choose from all of the modules that are enabled in the Profile Setup window.

'Find Address' Field
The 'Find Address' field allows the user to search for a particular address in this window.

To enter this field, move the mouse cursor to this field and then press the left mouse button (or press the "Tab" key until this field is highlighted). Once inside of the field, an address can be entered in hexadecimal or a symbol name can be entered by pressing the right mouse button (middle button on Unix systems) and then selecting the desired symbol. The address of that symbol will then be put into the 'Find Address' field. For additional information, see ***Working With Symbols*** in Chapter 2 of this manual.

Once the address or symbol name is entered, press the ENTER key to begin the search. If the address value is not found in the extended source data, the message "Out of range address" will be displayed.

PAC Extended Source Window Buttons
The buttons in the PAC Extended Source window allow the user to perform certain actions. For additional information, choose one of the following topics:

Close
The **Close** button will close the Extended Source window.

Help
The **Help** button will invoke help for the Extended Source window.

Scroll Box
The **Scroll Box** button will allow the user to scroll through the available choices for certain fields.

Printing PAC Information

PAC information can be printed to a printer or to a file by choosing 'File/Print'. The information that is printed depends on the window that is active when the print command is invoked. When you choose to print a window, the Print Output Selection Window will be shown. The following topics provide additional information on printing PAC information:

Printing Files From The Main PAC Window

If you choose 'File/Print' while the main PAC window is active (the one labeled "Performance Analysis"), you will then be prompted to enter a file to print. This file can be any file on your system that is printable. This feature can be handy if you have previously saved some PAC information to a file and you wish to obtain a hard copy of it.

Printing The PAC Setup

If you choose 'File/Print' while one of the PAC Setup windows is active, you can then print the PAC setup information.

The following example shows the type of information that will be shown:

```
Performance Analysis Setups
===========================
Runs Requested: Continuous
Cumulative: YES
Trigger Mode: Automatic
Start Module: Module #0
Stop Module: Module #0
Requested Record Time: 60000 ms
---------------------------------
Module #0: init
Entry-Exit: 00000000H-00000000H
Address-Range: 000005F8H-00000612H
---------------------------------
Module #1: marquee
Entry-Exit: 00000000H-00000000H
Address-Range: 00000400H-000004DCH
---------------------------------
etc....
```

Printing PAC Profile Data

If you choose 'File/Print' while one of the PAC Profile windows is active, you can then print the PAC Profile information.

The following example shows the type of information that will be shown:

```
Thu Sep 29 16:18:20 1994
Performance Analysis Results
============================
Runs Requested: Continuous
Cumulative: YES
Trigger Mode: Automatic
Start Module: Module #0
Stop Module: Module #0
Requested Record Time: 60000 ms
Recording Time: 02.525 ms
Total Recording Time: 02.525 ms
Timer Resolution: 50 ns
Runs Executed: 1

--------------------------------
Module #0: init
Entry-Exit: 00000000H-00000000H
Address-Range: 000005F8H-00000612H
Executions: 2
Min. Time: 00.250 us
Max. Time: 00.250 us
Avg. Time: 00.250 us
Pct. Time: 000.020 %
Percent Covered: 28.57 %

--------------------------------
Module #1: marquee
Entry-Exit: 00000000H-00000000H
Address-Range: 00000400H-000004DCH
Executions: 1382
Min. Time: 00.100 us
Max. Time: 02.450 us
Avg. Time: 01.200 us
Pct. Time: 065.690 %
Percent Covered: 72.07 %

--------------------------------
etc...
```

Printing PAC Extended Coverage Data

If you choose 'File/Print' while the PAC Extended Coverage window is active, you can then print the PAC Extended Coverage information. See Select Coverage Range Window for additional information on how to print from this window.

The following example shows the type of information that will be shown. Note that - means that the line was not covered, + means that the line was covered.

```
Thu Sep 29 16:19:08 1994
Module: marquee
Address-Range: 00000400H-000004DCH
00000400h:-  00000402h:-  00000404h:-  00000406h:-  00000408h:-
0000040Ah:-  0000040Ch:-  0000040Eh:-  00000410h:-  00000412h:-
00000414h:-  00000416h:+  00000418h:-  0000041Ah:-  0000041Ch:-
0000041Eh:-  00000420h:-  00000422h:-  00000424h:-  00000426h:-
00000428h:-  0000042Ah:-  0000042Ch:-  0000042Eh:-  00000430h:-
00000432h:-  00000434h:-  00000436h:-  00000438h:+  0000043Ah:+
0000043Ch:+  0000043Eh:+  00000440h:+  00000442h:+  00000444h:+
00000446h:+  00000448h:+  0000044Ah:+  0000044Ch:+  0000044Eh:+
00000450h:+  00000452h:+  00000454h:+  00000456h:+  00000458h:+
0000045Ah:+  0000045Ch:+  0000045Eh:+  00000460h:+  00000462h:+
00000464h:+  00000466h:+  00000468h:+  0000046Ah:+  0000046Ch:+
0000046Eh:+  00000470h:+  00000472h:+  00000474h:+  00000476h:+
00000478h:+  0000047Ah:+  0000047Ch:+  0000047Eh:+  00000480h:+
00000482h:+  00000484h:+  00000486h:+  00000488h:+  0000048Ah:+
0000048Ch:+  0000048Eh:+  00000490h:+  00000492h:+  00000494h:+
00000496h:+  00000498h:+  0000049Ah:+  0000049Ch:+  0000049Eh:+
000004A0h:+  000004A2h:+  000004A4h:+  000004A6h:+  000004A8h:+
000004AAh:+  000004ACh:+  000004AEh:+  000004B0h:+  000004B2h:+
000004B4h:+  000004B6h:+  000004B8h:+  000004BAh:+  000004BCh:+
000004BEh:+  000004C0h:+  000004C2h:+  000004C4h:+  000004C6h:+
000004C8h:-  000004CAh:-  000004CCh:-  000004CEh:-  000004D0h:+
000004D2h:+  000004D4h:+  000004D6h:+  000004D8h:+  000004DAh:+
Note:
+ ==> Address Covered
- ==> Address Not Covered
```

Printing PAC Extended Source Data

If you choose 'File/Print' while the PAC Extended Source window is active, you can then print the PAC Extended Source information. This window will show covered code in either source (high-level) or assembly language format. See Select Coverage Range Window for additional information on how to print from this window.

The following example shows the type of information that will be shown if the window is in "Code" format. Note that # shows that the line generated no code (comment), - means that the line was not covered, + means that the line was covered.

```
Thu Sep 29 16:19:36 1994
Module: marquee
Address-Range: 00000400H-000004DCH
00000400h #
00000408h #     int     n,i;
00000418h -     unsigned int msg_switch=2;
00000418h #     static long  delay_val=DELAY;
00000418h #
00000418h #
00000418h # /* clear screen and initialize the marquee data */
00000428h -     for (n=0; n<WIDTH; n++)
0000041Ch -             marquee[n]='.';
0000041Ch #
00000430h -     demo_msg.length = WIDTH;
00000430h #
00000438h +     i=0;
... etc ...
=======================================
Note:
+ ==> Address Covered
- ==> Address Not Covered
```

The following example shows the type of information that will be shown if the window is in "Disassembly" format. Note that - means that the line was not covered, + means that the line was covered.

```
Thu Sep 29 16:20:03 1994
Module: marquee
Address-Range: 00000400H-000004DCH
00000400h - LINK.W A6,#FFF4
00000404h - MOVEM.L D2-D4,-(A7)
00000408h - LEA (FFF4,A6),A0
0000040Ch - MOVEA.L #00031000,A1
00000412h - MOVE.L (A1)+,(A0)+
00000414h - MOVE.L (A1)+,(A0)+
00000416h + MOVE.L (A1)+,(A0)+
00000418h - MOVEQ #02,D2
0000041Ah - MOVEQ #00,D3
0000041Ch - MOVEA.L #00030000,A0
00000422h - MOVE.B #2E,(00,A0,D3.W)
00000428h - ADDQ.L #1,D3
0000042Ah - MOVEQ #24,D0
0000042Ch - CMP.L D3,D0
0000042Eh - BGT.S 0000041C
00000430h - MOVEQ #24,D0
00000432h - MOVE.L D0,0003002C.L
00000438h + MOVEQ #00,D4
0000043Ah + JSR.L 000004DE
00000440h + MOVEQ #02,D0
... etc ...
=======================================
Note:
+ ==> Address Covered
- ==> Address Not Covered
```

Appendix

This Appendix contains other items of interest relating to the use of SourceGate II and your HMI Development system. For the latest information, please consult your on-line help from within SourceGate II or contact HMI Technical Support.

Error Messages

The following is an overview of some of the more common error messages that could be encountered when using SourceGate II. If you encounter an error message that is not listed here, contact HMI Technical Support.

All Systems

The following error messages may be encountered when using any HMI Development System.

ERROR:
WARNING! Cannot open the Configuration File:XXXXX.SG. Please Check to see that this File is on Disk in the Current Directory or in the Search Path. If this file has been renamed, Please enter the New Name.

Enter Configuration File Name:

CAUSE:
The personality file for your particular system could not be found in the current directory or in the search path

SOLUTION:
Move the named file to your current directory, change to the directory where this file resides, or include the directory where this file resides in your search path. Then, press the "OK" button. You can also use the "Browse" button to search for the needed file on your computer or you can type the full path name for the file in the box provided.

ERROR:
WARNING! Cannot open the Configuration File:XXXXX.HLP. Please Check to see that this File is on Disk in the Current Directory or in the Search Path. If this file has been renamed, Please enter the New Name.

Enter Configuration File Name:

CAUSE:
The Windows help file for your particular system could not be found in the current directory or in the search path

SOLUTION:
Move the named file to your current directory, change to the directory where this file resides, or include the directory where this file resides in your search path. Then, press the "OK" button. You can also use the "Browse" button to search for the needed file on your computer or you can type the full path name for the file in the box provided.

ERROR:
WARNING! Cannot open the Configuration File:XXXXX.CSC. Please Check to see that this File is on Disk in the Current Directory or in the Search Path. If this file has been renamed, Please enter the New Name.

Enter Configuration File Name:

CAUSE:
The Unix help file for your particular system could not be found in the current directory or in the search path

SOLUTION:
Move the named file to your current directory, change to the directory where this file resides, or include the directory where this file resides in your search path. Then, press the "OK" button. You can also use the "Browse" button to search for the needed file on your computer or you can type the full path name for the file in the box provided.

ERROR:
There Are No Source Code Modules

CAUSE:
You attempted to perform a source-level debugging operation. However, a file containing source-level debugging information has not yet been loaded.

SOLUTION:
Load a file that contains source-level debugging information and try again.

ERROR:
There Are No Symbols in the Symbol Table

CAUSE:
You attempted to perform an operation that referenced a symbol. However, a file containing symbolic debugging information has not yet been loaded.

SOLUTION:
Load a file that contains symbolic debugging information and try again.

ERROR:
Unable To Open File: 'xxxxx.xxx'

CAUSE:
SourceGate II cannot find the file specified

SOLUTION:
Make sure that the filename and path specified are correct.

ERROR:
No CodeView Window Present For Tracking

CAUSE:
You tried to invoke a CodeView Cursor Tracking Assembly window when no CodeView window was open

SOLUTION:
Open a CodeView window and then invoke the Cursor Tracking Assembly window, or use the VASM command to open an assembly window. The VASM command allows you to open an assembly window at any time beginning at the defined address.

SPS-2000 Series/HMI-200 Series Emulators

The following error messages may be encountered when using the above systems.

ERROR:
"<hostname>" Refused To Connect. You Can: See Who is connected, Reset the emulator, or Exit SourceGate.

CAUSE:
You attempted to log onto an emulator via Ethernet that is currently in use by someone else.

SOLUTION:
You can press the "See Who" button to see who in your organization is currently logged onto the emulator via Ethernet. If you wish to kick the current user off of the system, press the "Reset" button. This will reset the Ethernet connection to the emulator, re-boot the emulator, and log you in via Ethernet. Note that the previous user will receive no warning about being kicked off of the emulator.

ERROR:
The Communications Software Has Failed to Receive A Response From The Emulator. Please Check The Communication Link, Or Try Resetting The Emulator.

CAUSE:
SourceGate II could not establish communications with the emulator.

SOLUTION:
This problem can be caused by several conditions. First, make sure that the emulator is powered on. The green LED on the front of the emulator should be on and the emulator fan should be running (HMI-200 Series and SPS-2000 units only). If it is not, check the power connection to the emulator. Second, make sure that a straight-through type of RS-232 cable is plugged into the MAIN serial port of the emulator on one end, and to a valid serial port on the computer end. Third, make sure that SourceGate is being invoked with the proper COM port setting as outlined in the section Configuring The Software. Finally, if your emulator has a pod with active circuitry on it, there will be a power cable that plugs into this pod. Make sure that this connection has been made. After making the above inspections, press the Retry button. If you still have problems after trying the above fixes, call HMI Technical Support.

ERROR:
Warning: PC and SP values are invalid

CAUSE:
Certain emulators, most notably the 68000, 68020, 68030, 68040, 68302, 68306, and 68300 family, fetch their reset and stack pointer vectors from memory upon power-up. If this memory is mapped to the emulator, the values fetched for these vectors will be random garbage since the processor is fetching values from uninitialized RAM.

SOLUTION:
Edit memory locations 0 - 7 so that they contain valid reset and stack pointer vectors and then perform the "RS" command from the Command window, or press the "RESET" button in the Run Status window.

Background Mode Debuggers (BMDs)

The following error messages may be encountered when using the above system.

ERROR:
Fatal Error From CFSC: BMD not responding

CAUSE:
SourceGate II could not establish communications with the BMD

SOLUTION:
This error can be caused by several things. First, make sure that the BMD is plugged into the debug port on the target system. Second, make sure that the 25 pin RS-232 connector on the BMD is connected to a straight-thru cable which should be connected to a serial port on your PC. Third, make sure that you have invoked the correct COM port when you started SourceGate. Forth, make sure that the target system has adequate power to drive the BMD (>250ma @ +5V) and that the red stripe on the BMD cable is aligned with pin 1 on the target system debug port connector. For additional information, see Getting Started With HMI Products and Using HMI BMDs With 3.3V Target Systems. If you still have problems, contact Technical Support.

ERROR:
Fatal Error From CFSC: Processor not supported by this BMD hardware

CAUSE:
The wrong CFSC_BMD.DLL file is being used

SOLUTION:
This error is caused by attempting to use the wrong CFSC_BMD.DLL file with the BMD. This could have happened during the installation process when you were prompted as to which BMD you wanted to install for. Re-install the software to see if this problem corrects itself or you can manually copy the correct .DLL file. To do this, delete the current CFSC_BMD.DLL file and then copy one of the supplied .DLL files (BMD332.DLL, BMD360.DLL, etc.) to the CFSC_BMD.DLL file. If you have questions on how to do this, contact Technical Support.

ERROR:
Cannot find CFSC_BMD.DLL, Windows needs this file to run SG2BMD.EXE

CAUSE:
The CFSC_BMD.DLL file is missing.

SOLUTION:
This error is caused by attempting to execute the SourceGate II BMD executable without the CFSC_BMD.DLL file being present. Re-install the software to see if this problem corrects itself or you can manually copy the correct .DLL file. To do this, copy one of the supplied .DLL files (BMD332.DLL, BMD360.DLL, etc.) to the CFSC_BMD.DLL file. If you have questions on how to do this, contact Technical Support.

Using Compilers and Assemblers

SourceGate is compatible with a number of downloadable file formats. SourceGate can load the code, data, symbolic, and source-level debugging information (line numbers) from those formats that supply such information.

SourceGate supports the following formats (those that do not produce symbol information (including line numbers) are so noted). Others may be supported. Consult HMI Technical Support if your file format does not appear here and you have trouble downloading your file.

- Motorola S-Record *
- Intel Hex *
- Tekhex *
- HMI Binary *
- Common Object File Format (COFF)
- Software Development Systems (SDS)
- IAR Systems
- IEEE-695
- ELF/DWARF
- Extended Tekhex
- Microtec Research Absolute
- Hitachi

  * - No symbol information (including line numbers) supported in this format

Most compiler/assembler manufacturers provide linkers that can produce one of the above formats. Other formats can be converted to Extended Tekhex with the object file converters supplied with your SourceGate software. Supplied on the converter disk are executable (.EXE or .COM) and documentation (.DOC) files for each converter listed below. Please read the applicable documentation file for the converter you wish to use before running the converter. The documentation files have been written to provide insight on how to use the converter programs properly.

Available File Converters

The following converters are available from HMI:

180AD2XT	2500 A.D. 64180 symbol table file to XTEKHEX file converter
2500TOXT	2500 A.D. standard symbol table to Extended TEKHEX file converter
68KAD2XT	2500 A.D. MC68000 symbol table file to Extended TEKHEX file converter
AATOXT	American Automation hex file to Extended TEKHEX file converter
ARCH2BNK	Archimedes banked "S" record splitter
ATOXT	UNIX A.OUT file to Extended TEKHEX file converter
AVOXTEK	AVOCET plain dump symbol file to Extended TEKHEX file converter
COFF2XT	COFF file (Introl, Sierra, Intermetrics, SUN386, and PG52x0 COFF) to Extended TEKHEX file converter
COFF2HMI	COFF file (Introl, Sierra, Intermetrics, SUN386, and PG52x0 COFF) to HMI Binary file converter
COMFORXT	COMFOR file to Extended TEKHEX file converter
DBGXTEK	Manx debug (DBG) file to Extended TEKHEX file converter
HIC2IAR	Hi-Cross (New Archimedes) to IAR (Old Archimedes) file converter
HTOXT	HITACHI hex file to Extended TEKHEX file converter
IAR2XT	IAR Systems UBROF Debug file to Extended TEKHEX file converter
IEEETOXT	IEEE 695 Object format file to Extended TEKHEX file converter
IEEE2IAR	IEEE 695 Object format file to IAR Systems file converter
ITOB	INTEL hex file to BINARY file converter
ITOM	INTEL hex file to MOTOROLA 'S' record file converter
MTTOXT	MICROTEK symbol table to Extended TEKHEX file converter
NWISTOXT	Northwest Instruments Symbol file to Extended TEKHEX file converter
OBJCNV	INTEL OMF Object file converter
OS92HMI	Microware OS9 file to SDS file converter
QUEXTEK	QUELO linker report file to Extended TEKHEX file converter
UTILXTEK	Intermetrics (Whitesmiths) Object file to Extended TEKHEX file converter. Note: A special converter is available for the Whitesmith 68HC11. Contact the factory for additional information.
ZTOXT	INTEL hex file/ZILOG symbol file to Extended TEKHEX file converter

Index

A

A (In-Line Assembly), 4-5
Accelerator Keys, 2-17
Appendix, A-1
Assemblers, Using, A-7

B

BM (Byte Mode), 4-5
BMD Interface, 4-1
Bus Activity, 3-50
Byte Order, 2-20

C

C++ Class Browser Window, 3-64
C++ Class Sub Browser Window, 3-65
C (Compare Memory Block), 4-6
Checking Your Package, 1-1
CM (Show Status Of Control Board Memory), 4-7
CodeView BPs Window, 5-29
CodeView Color Selection Window, 5-16
CodeView Functional Options Window, 5-18
Codeview Global View Parameters Window, 5-12
CodeView Interface, 5-1
CodeView Setup Window, 5-7
CodeView Window, Searching, 5-34
CodeView Window, Viewing Multiple, 5-22
CodeView Window Preferences, 5-7
CodeView Windows, Changing Display Mode, 5-25
CodeView Windows, Loading Source Code Into, 5-21
CodeView Windows, Setting Breakpoints In, 5-28
CodeView Windows, Single-Stepping In, 5-31
CodeView Windows, Starting Emulation From, 5-28
CodeView Windows, Viewing Assembly Code In, 5-23
Command File, 2-2
Command file, 2-3
Command Files, 3-8
Command Line Options, 1-9
Command Line Overrides, 3-3, 4-2
Command Window, 2-3, 3-1, 4-1
Command Window Commands, 3-2
Compiler Interface Options, 2-19
Compilers, Using, A-7
Cursor Tracking Assembly Window, 5-33

D

D (Dump Memory Block), 4-7
Disk Space Requirements, 1-2
DM (Data Memory Mode), 4-7
Download Object File, 2-2
Download Options, 2-26
Downloading An Object File, 2-25

E

E/EN (Enter/Change Memory), 4-8
Emulator Status Window, 3-9
Environment Variables, 1-13
Error Messages, A-1
Exiting SourceGate II, 2-3, 2-17
EXP (Examine/Change Peripheral), 4-9
EXR (Examine/Change Peripheral Register), 4-9

F

F (Fill Memory Block), 4-10
Filtered, 3-50
Floating Point Format, 2-20
Freeze Trace, 3-53

G

G (Begin Real-Time Emulation), 4-11
General Overview, 2-1
Getting Started, 1-1
Go From/To Widow, 3-59

H

Hardware Connections, 1-5
Hardware Requirements, 1-2
Help System, 2-10, 2-15
HLD (Hex Listing Disabled), 4-12
HLE (Hex Listing Enabled), 4-12

I

I/IN (Input), 4-12
Installing SourceGate II, 1-3

L

L (List Disassembled Code), 4-13
Line Symbol Format, 2-19
LM (Long Word Display Mode), 4-13
Log File, 2-3

M

M (Move Memory Block), 4-14
Memory Map, 2-7
Memory Output Window, 3-31
Memory Requirements, 1-2
Memory Window, 3-12
Memory Window, Accessing I/O, 3-29
Memory Window, Changing Display Mode, 3-17
Memory Window, Comparing Memory Blocks, 3-21
Memory Window, Filling Blocks, 3-23
Memory Window, Modifying Specific Locations, 3-16
Memory Window, Moving Blocks, 3-25
Memory Window, Searching, 3-19
Memory Window, Testing Blocks, 3-27
Memory Window, Viewing Specific Locations, 3-15
Menu Bar, 2-2
Modifying Register Values, 3-58, 4-32
Module Selection Window, 2-4, 5-21
MT/MTL (Test Block Of Memory), 4-15

O

O/ON (Output), 4-16
Object File, 2-2, 2-25, 2-28
Object File Formats, 2-24
Object Files, 2-24

P

PAC, 6-1
PAC, Printing Information, 6-35
PAC, Recording Profile Data, 6-19
PAC, Saving and Restoring, 6-18
PAC, Starting A Coverage Profile Session, 6-26
PAC, Starting A Profile Session, 6-24
PAC, Stopping A Coverage Profile Session, 6-26
PAC, Stopping A Profile Session, 6-24
PAC, Viewing Coverage Data, 6-25
PAC Coverage Profile Window, 6-25
PAC Execution Profile Graph Window, 6-23
PAC Execution Profile Window, 6-19
PAC Extended Coverage Window, 6-27
PAC Extended Source Coverage Window, 6-31
PAC Profile Mode Setup Window, 6-12
PAC Profile Setup Window, 6-3
PAC Profile Setup Window Example, 6-8
PAC Setup, 6-3
PAC Setup Qualify Window, 6-15
Path, 3-6
PC Tracking Window, 5-32

Performance Analysis (PAC), 6-1
PM (Program Memory Mode), 4-16
Pointers, 2-20
Ports, 3-50
Preferences, 2-22, 5-7
Printer Selection, 2-12
Printer Setup, 2-3
Program Counter, 2-20
pSOS Window, 3-67

R

RD (Read Object File Into Memory), 4-17
Register Map Window, 3-63
Register Window, 3-57, 4-31
Register Window, Available Registers, 3-58
RP (Reset Processor, Continue Emulation), 4-18
RS (Reset Processor, Stop Emulation), 4-18

S

S_PMBA (Set Prog. Module Base Address), 4-19
S_ROMWIDTH (Set ROM Width), 4-20
Search Paths, 3-6
Serial Interface Parameters, 2-21
Serial Port Parameters, 2-7
Shortcut Keys, 2-17
SM (Supervisor Memory Mode), 4-24
Software Requirements, 1-2
SourceGate Windows, 3-1
SP (Stop Emulation), 4-24
SR/SRN (Search Block Of Memory), 4-25
SS (Single Step), 4-26
ST (Show Emulator Status), 4-28
Stack Frame Pointer, 2-20
Stack Pointer, 2-20
Stack Trace Window, 3-61
Starting SourceGate II, 1-7
Status Bit, 3-49
Symbol Selection Window, 2-5, 2-12
System Requirements, 1-2

T

Time Stamp, 3-50
Trace Bus Activity, 3-50
Trace Display Fields, 3-48
Trace Status Bit, 3-49
Trace Time Stamp, 3-50
Trace Window, 3-46
Trace Window, Capturing, 3-47
Trace Window, Choosing Buffer, 3-47
Trace Window, Display Format, 3-51
Trace Window, Field Description, 3-48
Trace Window, Freezing/Resuming, 3-53
Trace Window, Saving Information, 3-55
Trace Window, Searching, 3-54
Trace Window, Using, 3-46
Trace Window Buttons, 3-56

U

UM (User Memory Mode), 4-28
Upload Object File, 2-2
Uploading An Object File, 2-28
Using Compilers and Assemblers, A-7
Using SourceGate Windows, 3-1

V

VASM (View Assembly), 4-28
VER (Display Firmware Version), 4-28

W

W (Write Memory Contents To File), 4-29
Watch In Real Time Window, 3-33
Watch Variable, Adding, 3-35
Watch Variable, Deleting, 3-43
Watch Variable, Modifying, 3-43
Watch Variable Addresses, 3-37
Watch Variable Display Types, 3-37
Watch Variable Overrides, 3-37

Watch Variable Types, 3-36
Watch Window, 3-33
Watch Window, Restoring, 3-43
Watch Window, Saving, 3-43
Watch Window, Using Overrides, 3-45
Watch Window Examples, 3-43
Watching A Multi-Dimensional Array, 3-44
Watching A One-Dimensional Array, 3-44
Watching A Structure, 3-43
Watching a Symbol, 3-35
Watching An Element Of A Structure, 3-44
Watching Pointers, 3-45
Window Manipulation, 2-11
WM (Word Display Mode), 4-30

X

X (Examine/Change Registers), 4-30